NAME : PAREENITA A.SHIRSATH PRN : 221101062 ROLL.NO : 57 T.E.A.I.&.D.S.

**ML EXPERIMENT NO : 4**

**CODE:**

1)import numpy as np

import pandas as pd

import seaborn as sns

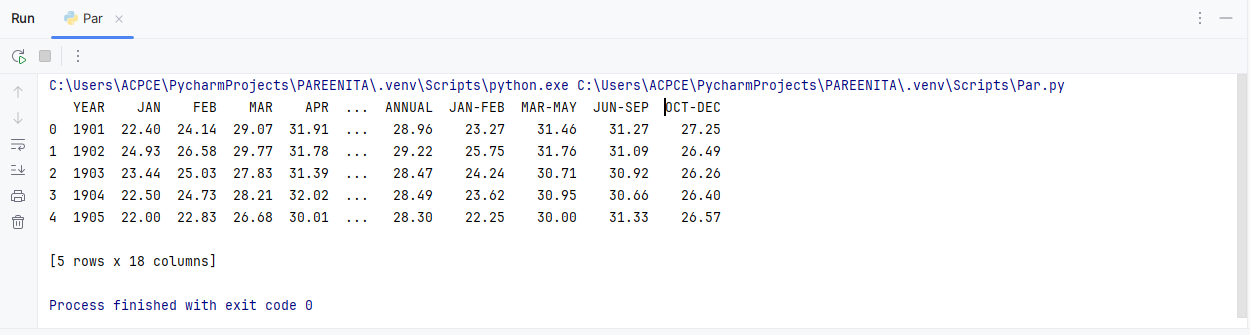
from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

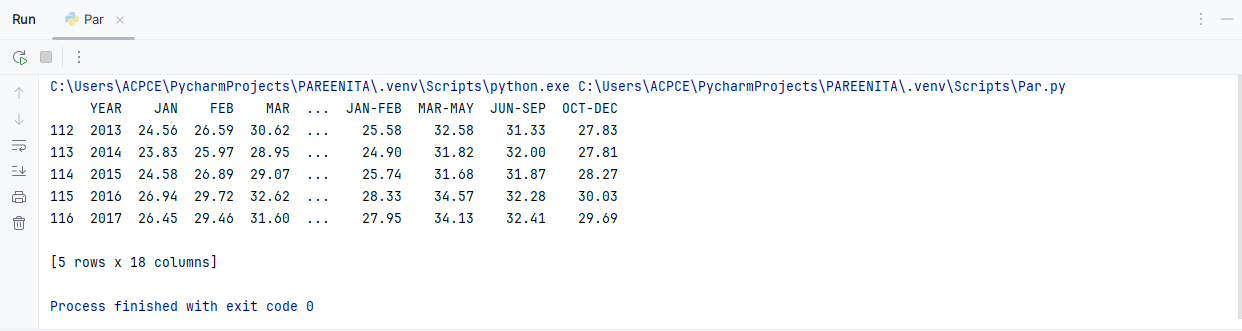
import matplotlib.pyplot as plt

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

2)print(data.head())



3)print (data.tail())



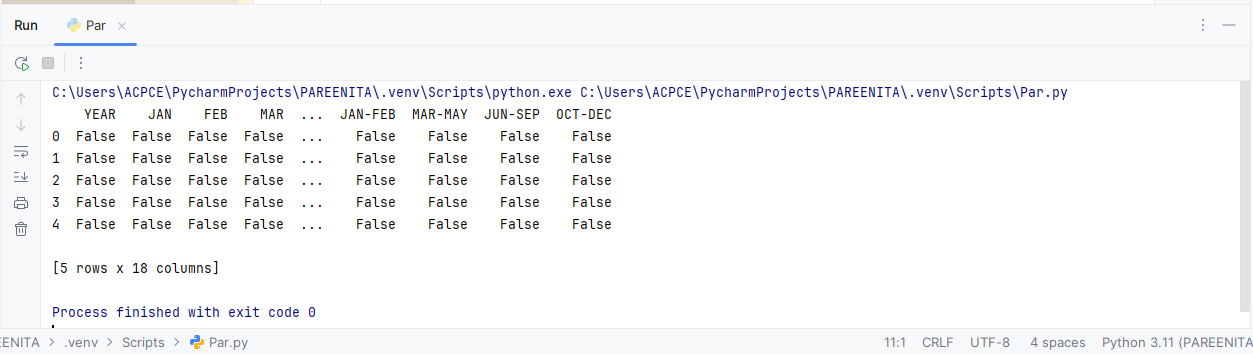
4)print(data.shape)



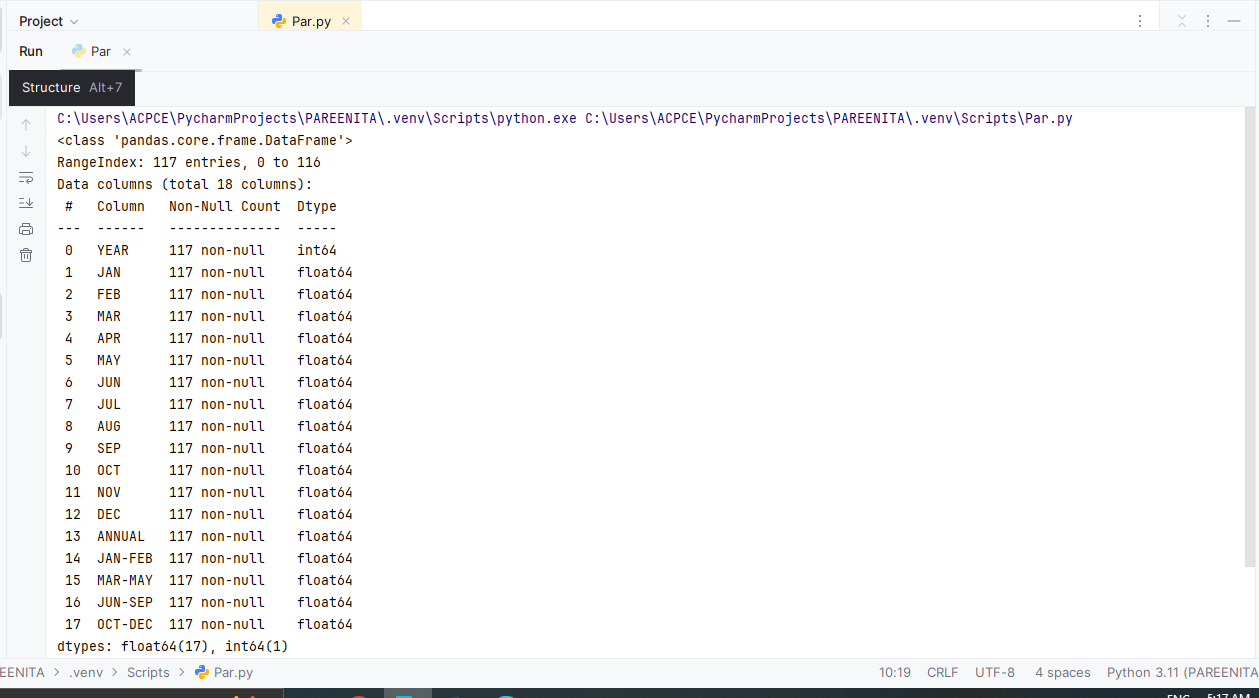
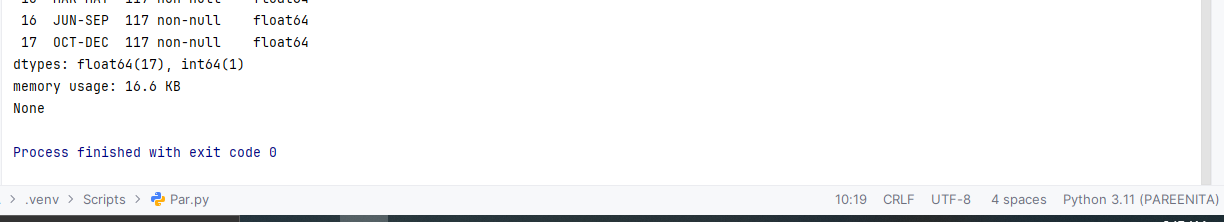
5)print(data.isnull().sum())



6)print(data.isnull().head())



7)print(data.info())

8)import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

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

# Load data

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

# Extract features and target variable

**x = data["YEAR"]**

**y = data["ANNUAL"]**

# Split the data into training and test sets

**x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=0)**

# Create the scatter plot

**plt.figure(figsize=(10,6))** # Optional: Specify figure size

plt.title("Temperature Plot of India")

plt.xlabel("Year")

plt.ylabel("Temperature")

plt.scatter(x, y, color='blue', label='Full Data') # Scatter plot for full data

# Optionally, plot the training data as well

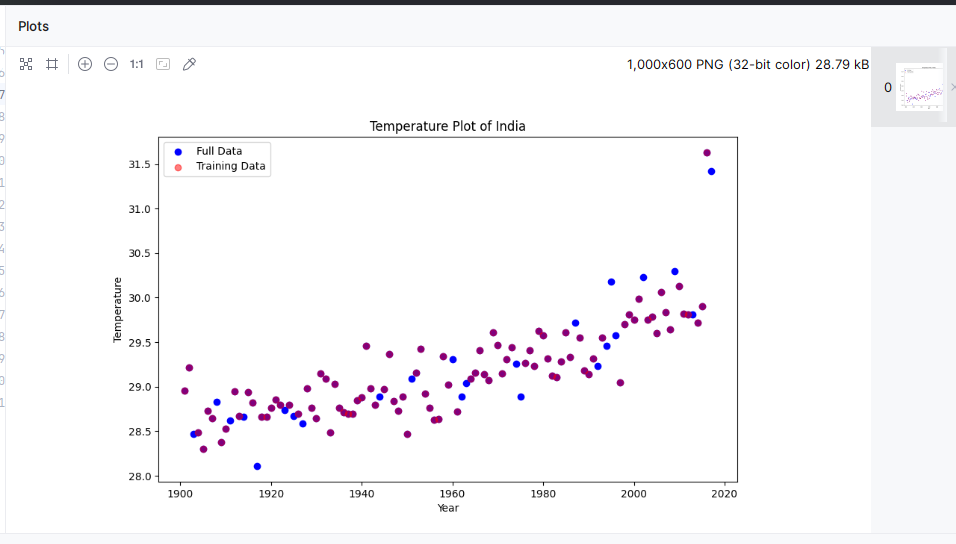
plt.scatter(x\_train, y\_train, color='red', label='Training Data', alpha=0.5)

# Show the legend to distinguish the points

plt.legend()

# Display the plot

plt.show()



9)# Print shapes of the training and test sets

print("Shape of x\_train:", x\_train.shape)

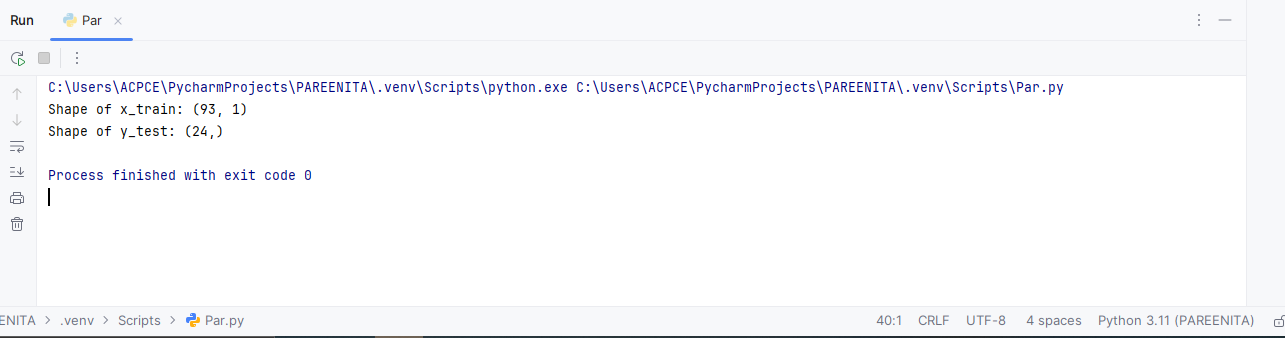
print("Shape of y\_test:", y\_test.shape)



10)x = data["YEAR"].values.reshape(117, 1)

x\_train = x\_train.reshape(97, 1)

x\_test = x\_test.reshape(20, 1)



11)x = data["YEAR"].values # No reshape here, just the raw values

y = data["ANNUAL"]

# HIGHLIGHTED: Print the shape of x to inspect the number of rows in x

print("Shape of x:", x.shape) # To inspect the shape of the original x

# Split the data into training and test sets

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=0)

# HIGHLIGHTED: Reshape x\_train and x\_test (automatically according to their sizes)

x\_train = x\_train.reshape(-1, 1) # Automatically reshape based on the size of x\_train

x\_test = x\_test.reshape(-1, 1) # Automatically reshape based on the size of x\_test

# HIGHLIGHTED: Print shapes of the training and test sets

print("Shape of x\_train:", x\_train.shape)

print("Shape of y\_test:", y\_test.shape)



`2)print("Data loaded successfully. Columns:", data.columns)

# Ensure the column "YEAR" exists in the dataframe

if "YEAR" in data.columns:

x = data["YEAR"].values # No reshape here, just the raw values

else:

raise ValueError("Column 'YEAR' not found in the CSV file")

y = data["ANNUAL"]

# HIGHLIGHTED: Print the shape of x to inspect the number of rows in x

print("Shape of x:", x.shape) # To inspect the shape of the original x

# Split the data into training and test sets

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=0)

# HIGHLIGHTED: Reshape x\_train and x\_test (automatically according to their sizes)

x\_train = x\_train.reshape(-1, 1) # Automatically reshape based on the size of x\_train

x\_test = x\_test.reshape(-1, 1) # Automatically reshape based on the size of x\_test

# HIGHLIGHTED: Print shapes of the training and test sets

print("Shape of x\_train:", x\_train.shape)

print("Shape of y\_test:", y\_test.shape)

# HIGHLIGHTED: Create the Linear Regression model and fit it to the training data

model = LinearRegression()

model.fit(x\_train, y\_train) # Fit the model with training data (x\_train, y\_train)

# model.fit(x, y) # This line has been removed, as requested

