Sample Program1:

-> Simple Linear Regression Model….

def:

Linear regression is a type of supervised machine learning algorithm that computes the linear relationship between the dependent variable and one or more

Step1: Env Stepup,

```
-> python -m venv myenv
   source myenv/bin/activate
   pip install numpy pandas scikit-learn matplotlib
```

Step2: Create a Simple Dataset

Create a file named data.csv:

```
-> x,y
1,2
2,4
3,6
4,8
5,10
```

Step 3:Write a Python Script

Create a Python script named playground.py

```
->import matplotlib #import the matplotlib.
```

```
->matplotlib.use('Agg')  # Use Agg backend for a non-display driver
system. So it does not need a display.
```

```
->import numpy as np                      #import numpy(fundamental package
for scientific computing in Python)
```

```
->import pandas as pd             #import pandas(data manipulation and
analysis)
```

```python
->from sklearn.linear_model import LinearRegression #import class from the
scikit-learn library, which is used for linear regression modeling.

->import matplotlib.pyplot as plt #which is used for creating plots and
visualizations.



->data = pd.read_csv('data.csv')   # Load the dataset

# Prepare the data
->X = data[['x']].values  # Features (input)
->y = data['y'].values    # Target (output)



# Create and train the model
->model = LinearRegression()
->model.fit(X, y)  # linear regression model to the input data (X) and
target data (y). This trains the model to learn the relationship between X
and y.



# Make predictions
->predictions = model.predict(X) # trained model to predict the target
values based on the input data (X).

# Print the model parameters
->print(f'Coefficients: {model.coef_[0]}') #Prints the coefficient
->print(f'Intercept: {model.intercept_}') # Prints the intercept



# Plot the data and the model
->plt.scatter(X, y, color='blue', label='Actual data')
->plt.plot(X, predictions, color='red', linewidth=2, label='Fitted line')
->plt.xlabel('x')
->plt.ylabel('y')
->plt.title('Simple Linear Regression')
->plt.legend()

# Save the plot as a PNG file
->plt.savefig('plot.png')
```