Que: Create ‘sales’ Data set having 5 columns namely: ID, TV, Radio, Newspaper and Sales.(random 500 entries) Build a linear regression model by identifying independent and target variable. Split the variables into training and testing sets. then divide the training and testing sets into a 7:3 ratio, respectively and print them. Build a simple linear regression model.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

np.random.seed(42)
data = {
    'ID': np.arange(1, 501),
    'TV': np.random.randint(10, 100, 500),
    'Radio': np.random.randint(5, 50, 500),
    'Newspaper': np.random.randint(1, 20, 500),
    'Sales': np.random.randint(50, 200, 500)
}

df = pd.DataFrame(data)

# Split the data into features (X) and target variable (y)
X = df[['TV', 'Radio', 'Newspaper']]
y = df['Sales']

# 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)

print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"y_test shape: {y_test.shape}")

# Create a linear regression model
model = LinearRegression()

model.fit(X_train, y_train)
# Make predictions on the testing data
y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

Que:Create ‘realestate’ Data set having 4 columns namely: ID,flat, houses and purchases (random 500 entries). Build a linear regression model by identifying independent and target variable. Split the variables into training and testing sets and print them. Build a simple linear regression model for predicting purchases.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

np.random.seed(42)
data = {
    'ID': np.arange(1, 501),
    'flat': np.random.randint(10, 100, 500),
    'houses': np.random.randint(5, 50, 500),
    'purchases': np.random.randint(50, 200, 500)
}

df = pd.DataFrame(data)

# Split the data into features (X) and target variable (y)
X = df[['flat', 'houses']]
y = df['purchases']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# Print the shapes of the training and testing sets
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"y_test shape: {y_test.shape}")

# Create a linear regression model
model = LinearRegression()

model.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

Que: Create ‘User’ Data set having 5 columns namely: User ID, Gender, Age, EstimatedSalary and Purchased. Build a logistic regression model that can predict whether on the given parameter a person will buy a car or not.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

np.random.seed(42)
data = {
   'User ID': np.arange(1, 501),
   'Gender': np.random.choice(['Male', 'Female'], 500),
   'Age': np.random.randint(18, 60, 500),
   'EstimatedSalary': np.random.randint(15000, 150000, 500),
   'Purchased': np.random.choice([0, 1], 500)  # 0: Not Purchased, 1: Purchased
}

df = pd.DataFrame(data)

df['Gender'] = df['Gender'].map({'Male': 0, 'Female': 1})

# Split the data into features (X) and target variable (y)
X = df[['Gender', 'Age', 'EstimatedSalary']]
y = df['Purchased']

# 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)

model = LogisticRegression()

model.fit(X_train, y_train)
```

```
# Make predictions on the testing data
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print(f"Confusion Matrix:\n{conf_matrix}")
```