```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "execution_count": 1,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "Ft08J4hGhZxw",
        "outputId": "45d66c1c-5bf6-4184-e511-98e19cb119eb"
      },
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "Iris Dataset Accuracy: 1.0\n",
            "Classification Report:\n",
            "              precision    recall  f1-score   support\n",
            "\n",
            "           0       1.00      1.00      1.00        19\n",
            "           1       1.00      1.00      1.00        13\n",
            "           2       1.00      1.00      1.00        13\n",
            "\n",
            "    accuracy                           1.00        45\n",
            "   macro avg       1.00      1.00      1.00        45\n",
            "weighted avg       1.00      1.00      1.00        45\n",
            "\n",
            "Simulated Dataset Accuracy: 0.7111111111111111\n",
            "Classification Report:\n",
            "              precision    recall  f1-score   support\n",
            "\n",
            "           0       0.61      1.00      0.76        11\n",
            "           1       0.80      0.42      0.55        19\n",
            "           2       0.76      0.87      0.81        15\n",
            "\n",
            "    accuracy                           0.71        45\n",
            "   macro avg       0.73      0.76      0.71        45\n",
            "weighted avg       0.74      0.71      0.69        45\n",
            "\n"
          ]
        }
      ],
```

```
      "source": [
        "# Assignment 1: Machine Learning with Python\n",
        "\n",
        "# Step 1: Load and Understand the Iris Dataset\n",
        "\n",
        "import numpy as np\n",
        "import pandas as pd\n",
        "from sklearn import datasets\n",
        "from sklearn.model_selection import train_test_split\n",
        "from sklearn.preprocessing import StandardScaler\n",
        "from sklearn.linear_model import LogisticRegression\n",
        "from sklearn.metrics import accuracy_score,
classification_report\n",
        "\n",
        "# Load Iris dataset\n",
        "iris = datasets.load_iris()\n",
        "X_iris = iris.data\n",
        "y_iris = iris.target\n",
        "\n",
        "# Split the dataset into training and testing sets\n",
        "X_train_iris, X_test_iris, y_train_iris, y_test_iris =
train_test_split(X_iris, y_iris, test_size=0.3, random_state=42)\n",
        "\n",
        "# Standardize the features\n",
        "scaler = StandardScaler()\n",
        "X_train_iris = scaler.fit_transform(X_train_iris)\n",
        "X_test_iris = scaler.transform(X_test_iris)\n",
        "\n",
        "# Train a Logistic Regression model\n",
        "model_iris = LogisticRegression(max_iter=200)\n",
        "model_iris.fit(X_train_iris, y_train_iris)\n",
        "\n",
        "# Make predictions\n",
        "y_pred_iris = model_iris.predict(X_test_iris)\n",
        "\n",
        "# Evaluate the model\n",
        "accuracy_iris = accuracy_score(y_test_iris, y_pred_iris)\n",
        "print(\"Iris Dataset Accuracy:\", accuracy_iris)\n",
        "print(\"Classification Report:\\n\",
classification_report(y_test_iris, y_pred_iris))\n",
        "\n",
        "# Step 2: Create and Use a Simulated Dataset\n",
        "\n",
        "from sklearn.datasets import make_classification\n",
        "\n",
        "# Generate a simulated dataset\n",
        "X_sim, y_sim = make_classification(n_samples=150, n_features=4,
n_informative=3, n_redundant=0, n_classes=3, random_state=42)\n",
        "\n",
        "# Split the dataset into training and testing sets\n",
        "X_train_sim, X_test_sim, y_train_sim, y_test_sim =
train_test_split(X_sim, y_sim, test_size=0.3, random_state=42)\n",
        "\n",
        "# Standardize the features\n",
        "X_train_sim = scaler.fit_transform(X_train_sim)\n",
        "X_test_sim = scaler.transform(X_test_sim)\n",
        "\n",
        "# Train a Logistic Regression model\n",
```

```
        "model_sim = LogisticRegression(max_iter=200)\n",
        "model_sim.fit(X_train_sim, y_train_sim)\n",
        "\n",
        "# Make predictions\n",
        "y_pred_sim = model_sim.predict(X_test_sim)\n",
        "\n",
        "# Evaluate the model\n",
        "accuracy_sim = accuracy_score(y_test_sim, y_pred_sim)\n",
        "print(\"Simulated Dataset Accuracy:\", accuracy_sim)\n",
        "print(\"Classification Report:\\n\",
classification_report(y_test_sim, y_pred_sim))\n"
      ]
    }
  ]
}
```