Experiment-6---Heart-attack-prediction-using-MLP

Aim:

To construct a Multi-Layer Perceptron to predict heart attack using Python

Algorithm:

Step 1:

Import the required libraries: numpy, pandas, MLPClassifier, train_test_split, StandardScaler, accuracy_score, and matplotlib.pyplot.

Step 2:

Load the heart disease dataset from a file using pd.read_csv().

Step 3:

Separate the features and labels from the dataset using data.iloc values for features (X) and data.iloc[:, -1].values for labels (y).

Step 4:

Split the dataset into training and testing sets using train_test_split().

Step 5:

Normalize the feature data using StandardScaler() to scale the features to have zero mean and unit variance.

Step 6:

Create an MLPClassifier model with desired architecture and hyperparameters, such as hidden_layer_sizes, max_iter, and random_state.

Step 7:

Train the MLP model on the training data using mlp.fit(X_train, y_train). The model adjusts its weights and biases iteratively to minimize the training loss.

Step 8:

Make predictions on the testing set using mlp.predict(X_test).

Step 9:

Evaluate the model's accuracy by comparing the predicted labels (y_pred) with the actual labels (y_test) using accuracy_score().

Step 10:

Print the accuracy of the model.

Step 11:

Plot the error convergence during training using plt.plot() and plt.show().

Program:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score

data=pd.read_csv("heart.csv")
X=data.iloc[:, :-1].values #features
Y=data.iloc[:, -1].values  #labels

X_train,X_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=42)

scaler=StandardScaler()
X_train=scaler.fit_transform(X_train)
X_test=scaler.transform(X_test)

mlp=MLPClassifier(hidden_layer_sizes=(100,100),max_iter=1000,random_state=42)
training_loss=mlp.fit(X_train,y_train).loss_curve_

y_pred=mlp.predict(X_test)

accuracy=accuracy_score(y_test,y_pred)
print("Accuracy",accuracy)

plt.plot(training_loss)
plt.title("MLP Training Loss Convergence")
plt.xlabel("Iteration")
plt.ylabel("Training Losss")
plt.show()
```

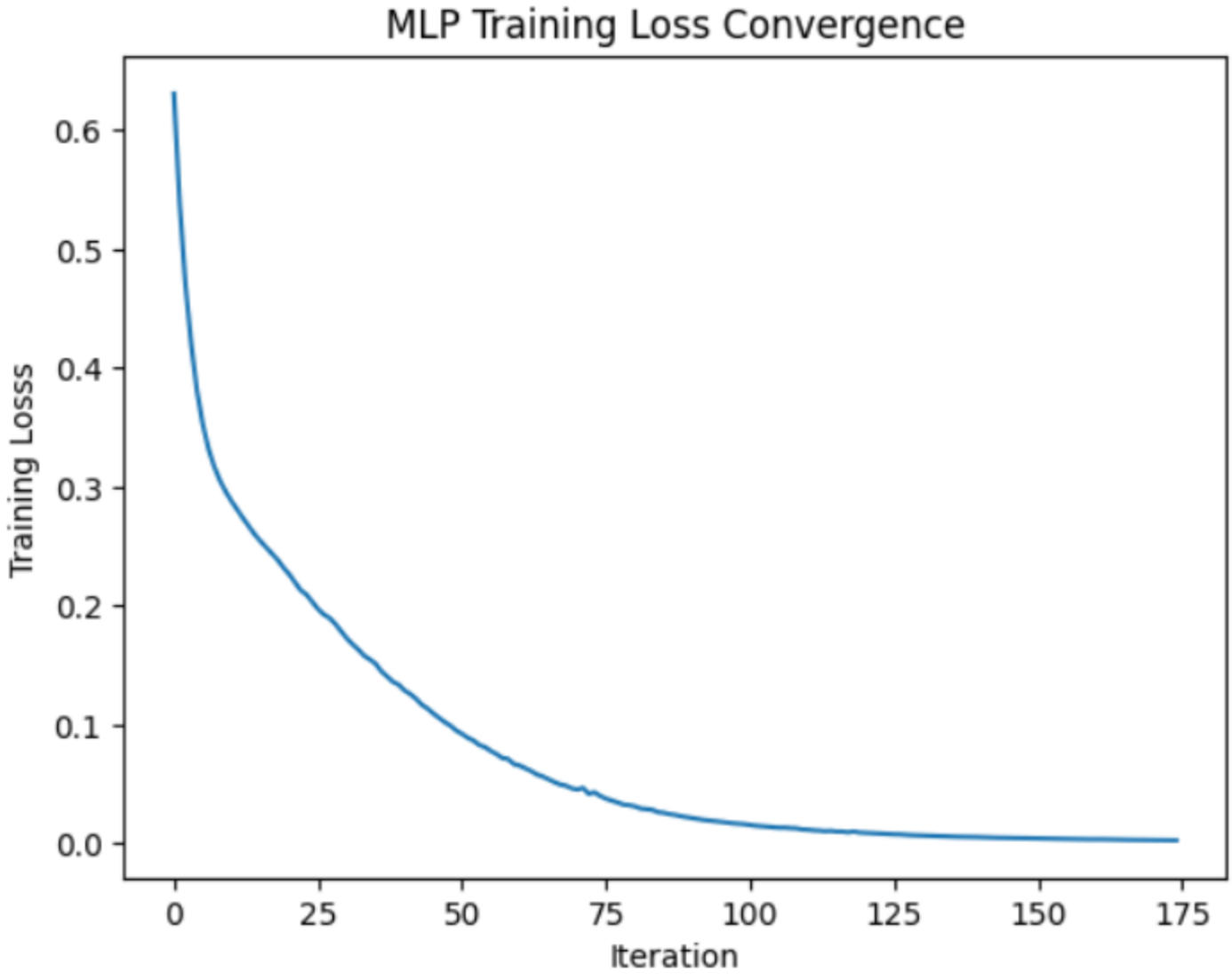
Output:

Accuracy:

```
accuracy=accuracy_score(y_test,y_pred)
print("Accuracy",accuracy)
```

```
Accuracy 0.9853658536585366
```

MLP Training Loss Convergence:



Result:

Thus, an ANN with MLP is constructed and trained to predict the heart attack using python.