A PYTHON PROGRAM TO IMPLEMENT MULTI LAYER PERCEPTRON WITH BACK PROPOGATION

Aim:

To implement multilayer perceptron with back propagation using python.

Algorithm:

Step 1: Import the Necessary Libraries

● Import pandas as pd.

● Import numpy as np.

Step 2: Read and Display the Dataset

● Use `pd.read_csv("dataset.csv")` to read the dataset.

● Assign the result to a variable (e.g., `data`).

● Display the first ten rows using `data.head(10)`.

Step 3: Display Dataset Dimensions

● Use the `.shape` attribute on the dataset (e.g., `data.shape`).

Step 4: Display Descriptive Statistics

● Use the `.describe()` function on the dataset (e.g., `data.describe()`).

Step 5: Import Train-Test Split Module

● Import `train_test_split` from `sklearn.model_selection`.

Step 6: Split Dataset with 80-20 Ratio

● Assign the features to a variable (e.g., `X = data.drop(columns='target')`).

● Assign the target variable to another variable (e.g., `y = data['target']`).

● Use `train_test_split` to split the dataset into training and testing sets with a ratio of 0.2.

● Assign the results to `x_train`, `x_test`, `y_train`, and `y_test`.

Step 7: Import MLPClassifier Module

- Import `MLPClassifier` from `sklearn.neural_network`.

Step 8: Initialize MLPClassifier

- Create an instance of `MLPClassifier` with `max_iter=500` and `activation='relu'`.
- Assign the instance to a variable (e.g., `clf`).

Step 9: Fit the Classifier

- Fit the model using `clf.fit(x_train, y_train)`.

Step 10: Make Predictions

- Use the `.predict()` function on `x_test` (e.g., `pred = clf.predict(x_test)`).
- Display the predictions.

Step 11: Import Metrics Modules

- Import `confusion_matrix` from `sklearn.metrics`.
- Import `classification_report` from `sklearn.metrics`.

Step 12: Display Confusion Matrix

- Use `confusion_matrix(y_test, pred)` to generate the confusion matrix.
- Display the confusion matrix.

Step 13: Display Classification Report

- Use `classification_report(y_test, pred)` to generate the classification report.
- Display the classification report.

Step 14: Repeat Steps 9-13 with Different Activation Functions

- Initialize `MLPClassifier` with `activation='logistic'`.
- Fit the model and make predictions.

● Display the confusion matrix and classification report.

● Repeat for `activation='tanh'`.

● Repeat for `activation='identity'`.

Step 15: Repeat Steps 7-14 with 70-30 Ratio

● Use `train_test_split` to split the dataset into training and testing sets with a ratio of 0.3.

● Assign the results to `x_train`, `x_test`, `y_train`, and `y_test`.

● Repeat Steps 7-14 with the new training and testing sets.