NAME: MADDIKUNTA DAKSHYANI

STUDENT ID: 700666204

CODE:

https://github.com/dxm62040ucm/Assignment5/blob/main/Assignment5.ipynb

VIDEO:

https://drive.google.com/file/d/12J1QPTBMmn1GGOZdfuWB_eDVMu5dTZD7/view?usp=sharing

Neural Networks & Deep Learning: ICP5

1) Implement Naïve Bayes method using scikit-learn library Use dataset available with name glass Use train_test_split to create training and testing part Evaluate the model on test part using score and classification_report(y_true, y_pred)

A) Firstly, I have read the given CSV File glass.csv using pandas read_csv method.

Printed the corresponding data frame using print statement.

```
[1]  #Read Glass CSV File
     import pandas as pd

     # Read the CSV file
     df = pd.read_csv('glass.csv')

     # Display the contents of the DataFrame
     print(df)
```

```
           RI     Na    Mg    Al     Si     K    Ca    Ba   Fe  Type
0    1.52101  13.64  4.49  1.10  71.78  0.06  8.75  0.00  0.0     1
1    1.51761  13.89  3.60  1.36  72.73  0.48  7.83  0.00  0.0     1
2    1.51618  13.53  3.55  1.54  72.99  0.39  7.78  0.00  0.0     1
3    1.51766  13.21  3.69  1.29  72.61  0.57  8.22  0.00  0.0     1
4    1.51742  13.27  3.62  1.24  73.08  0.55  8.07  0.00  0.0     1
..       ...    ...   ...   ...    ...   ...   ...   ...  ...   ...
209  1.51623  14.14  0.00  2.88  72.61  0.08  9.18  1.06  0.0     7
210  1.51685  14.92  0.00  1.99  73.06  0.00  8.40  1.59  0.0     7
211  1.52065  14.36  0.00  2.02  73.42  0.00  8.44  1.64  0.0     7
212  1.51651  14.38  0.00  1.94  73.61  0.00  8.48  1.57  0.0     7
213  1.51711  14.23  0.00  2.08  73.36  0.00  8.62  1.67  0.0     7

[214 rows x 10 columns]
```

Removed type column using drop method and that into X dataframe.

Splitting the given data into train and test.

20% of the data is stored in the test and 80% of data is stored in the train set.

Applied Naive bayes Model on train data frame.

Predicted the given model and printed accuracy and classification reports.

```
[34] from sklearn.naive_bayes import GaussianNB
     from sklearn.metrics import classification_report


     # Split the dataset into features and target variable
     X = df.drop('Type', axis=1)
     y = df['Type']
     print(X)
     print(y)
     # Split the data into training and testing sets
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4


     # Initialize the Naïve Bayes model
     nb_model = GaussianNB()

     # Train the model
     nb_model.fit(X_train, y_train)

     # Make predictions
     y_pred = nb_model.predict(X_test)
     print(y_pred)

     # Evaluate the model
     accuracy = nb_model.score(X_test, y_test)
     print("Accuracy:", accuracy)

     print(classification_report(y_test, y_pred))
```

2. Implement linear SVM method using scikit library Use the same dataset above Use train_test_split to create training and testing part Evaluate the model on test part using score and classification_report(y_true, y_pred) Which algorithm you got better accuracy? Can you justify why?

A) Same procedure is repeated for Linear SVM Model.

Printed accuracy and Classification report for Linear SVM Model.

Finally compared both the model's using accuracy and classification reports.

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report
from sklearn.svm import LinearSVC
from sklearn.svm import SVC
# Split the dataset into features and target variable
X = df.drop('Type', axis=1)
y = df['Type']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=
# Define and train the Linear SVM model
svm_model = SVC(kernel='linear', max_iter=10000)
svm_model.fit(X_train, y_train)


# Make predictions on the testing data
y_pred_svm = svm_model.predict(X_test)
print(y_pred_svm)
# Evaluate the SVM model
svm_accuracy = svm_model.score(X_test, y_test)
svm_classification_report = classification_report(y_test, y_pred_svm,  zero_division=1
```

```
# Compare accuracies
print("Linear SVM accuracy:", svm_accuracy)
print("Naive Bayes accuracy:", nb_accuracy)

# Compare classification reports
print("Linear SVM classification report:")
print(svm_classification_report)

print("Naive Bayes classification report:")
print(nb_classification_report)
```

From the given outputs, it is Clear that Accuracy of Linear SVm is far better than Naive Bayes Model.

```
Linear SVM accuracy: 0.7441860465116279
Naive Bayes accuracy: 0.5581395348837209
Linear SVM classification report:
              precision    recall  f1-score   support

           1       0.69      0.82      0.75        11
           2       0.67      0.71      0.69        14
           3       1.00      0.00      0.00         3
           5       0.80      1.00      0.89         4
           6       1.00      0.67      0.80         3
           7       0.88      0.88      0.88         8

    accuracy                           0.74        43
   macro avg       0.84      0.68      0.67        43
weighted avg       0.77      0.74      0.72        43

Naive Bayes classification report:
              precision    recall  f1-score   support

           1       0.41      0.64      0.50        11
           2       0.43      0.21      0.29        14
           3       0.40      0.67      0.50         3
           5       0.50      0.25      0.33         4
           6       1.00      1.00      1.00         3
           7       0.89      1.00      0.94         8

    accuracy                           0.56        43
   macro avg       0.60      0.63      0.59        43
weighted avg       0.55      0.56      0.53        43
```