ASSIGNMENT-5

NAME:-Bhanu chand Garikapati

STUDENT#:-700748274

GITHUB:- https://github.com/Bhanu5423/neural_assignments

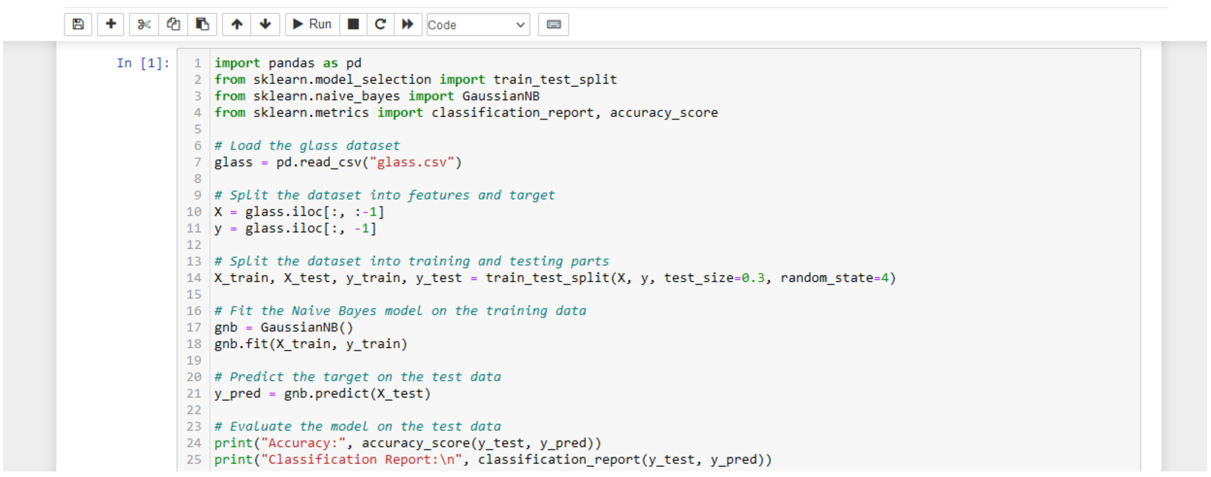
Q1. Implement Naïve Bayes method using scikit-learn library

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)



```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report, accuracy_score

# Load the glass dataset
glass = pd.read_csv("glass.csv")

# Split the dataset into features and target
X = glass.iloc[:, :-1]
y = glass.iloc[:, -1]

# Split the dataset into training and testing parts
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=4)

# Fit the Naive Bayes model on the training data
gnb = GaussianNB()
gnb.fit(X_train, y_train)

# Predict the target on the test data
y_pred = gnb.predict(X_test)

# Evaluate the model on the test data
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

1.Imported pandas library to read the data

2:From sklearn library I imported train test split

3.There are some classifiers in Naïve Bayes,

GuassianNB is one of the classifier.

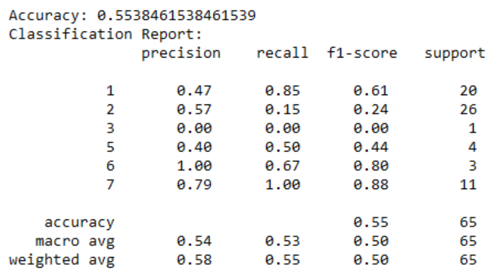
4.Data is read from glass.csv file and stored in glass.

5. x is input and y is output

6.We used train test split to create training and testing part.

7.After that, we evaluated the model on test part using

score card classification report(y_true, y_pred).



```
Accuracy: 0.5538461538461539
Classification Report:
              precision    recall  f1-score   support

           1       0.47      0.85      0.61        20
           2       0.57      0.15      0.24        26
           3       0.00      0.00      0.00         1
           5       0.40      0.50      0.44         4
           6       1.00      0.67      0.80         3
           7       0.79      1.00      0.88        11

    accuracy                           0.55        65
   macro avg       0.54      0.53      0.50        65
weighted avg       0.58      0.55      0.50        65
```

Output: Accuracy is 55%

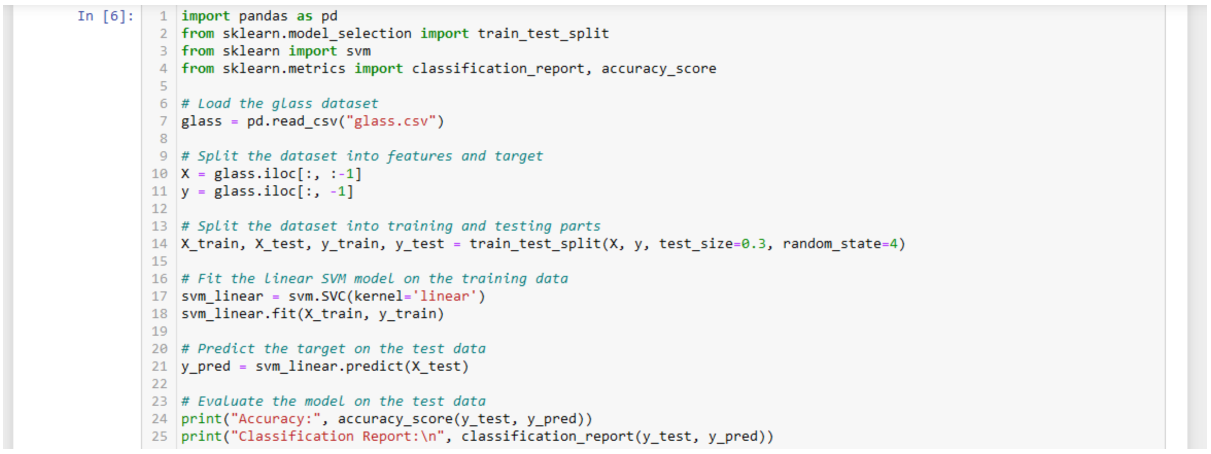
2. Implement linear SVM method using scikit library

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)



```
In [6]:
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import classification_report, accuracy_score

# Load the glass dataset
glass = pd.read_csv("glass.csv")

# Split the dataset into features and target
X = glass.iloc[:, :-1]
y = glass.iloc[:, -1]

# Split the dataset into training and testing parts
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=4)

# Fit the linear SVM model on the training data
svm_linear = svm.SVC(kernel='linear')
svm_linear.fit(X_train, y_train)

# Predict the target on the test data
y_pred = svm_linear.predict(X_test)

# Evaluate the model on the test data
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

1. Imported pandas library to read the data.

2. From sklearn library I imported train test split

3. There are some classifiers in SVM , SVC is one of the classifier.

4. Data is read from glass.csv file and stored in glass.

5. x is input and y is output

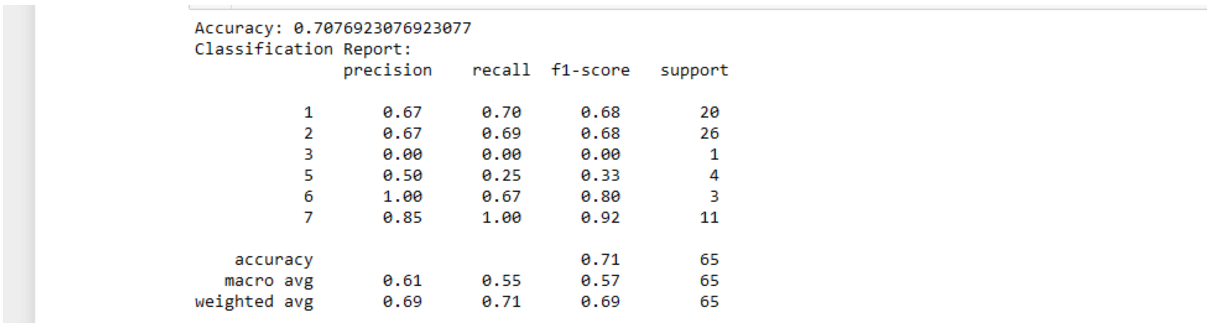
6. we used train test split to create training and testing part.

7. Fit the linear svm model on the training data

8. Predicted the target on the test data.

9. After that, we evaluated the model on test part using

 score card classification report(y_true, y_pred).



```
Accuracy: 0.7076923076923077
Classification Report:
              precision    recall  f1-score   support

           1       0.67      0.70      0.68        20
           2       0.67      0.69      0.68        26
           3       0.00      0.00      0.00         1
           5       0.50      0.25      0.33         4
           6       1.00      0.67      0.80         3
           7       0.85      1.00      0.92        11

    accuracy                           0.71        65
   macro avg       0.61      0.55      0.57        65
weighted avg       0.69      0.71      0.69        65
```

Output: Accuracy is 71%

Q: Which algorithm you got better accuracy? Can you justify why?

I achieved higher accuracy with the SVM algorithm due to its utilization of a geometric approach in data analysis. By default, SVM operates as a binary classifier, mapping data points in space to optimize the separation between two categories.