<u>ICP5</u>

Name: Vamsi Krishna Remala

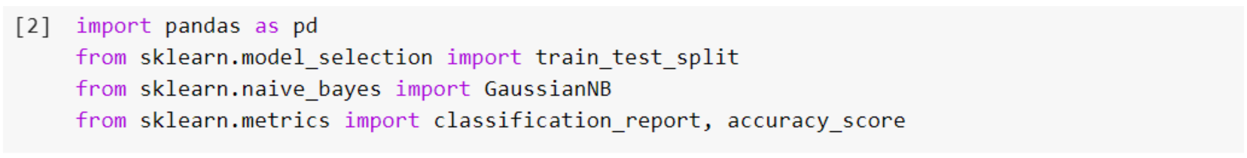
Student id: 700744730

GitHub Link:
https://github.com/vamsikrishnaremala/700744730_NNDL_ICP5

Video Link:
https://drive.google.com/file/d/1PbpwqblkGahqDzB_zy5cCZcD5inZIN-3/view?usp=drive_link

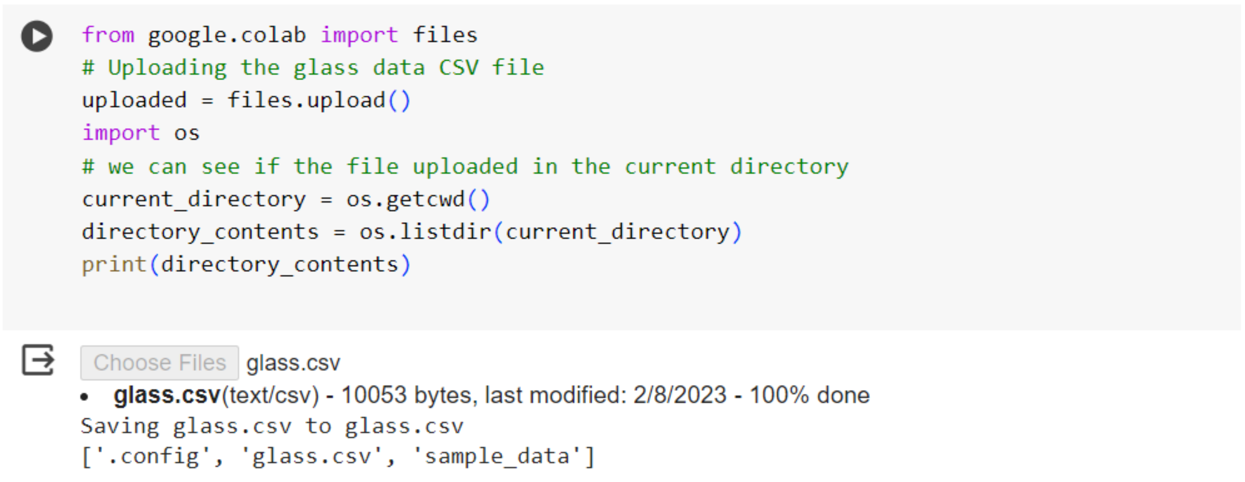
1. Implement Naïve Bayes method using scikit-learn library
   Use dataset available with name glass
   Use train_test_split to create training and testing part
   Evaluate the model on test part using score and classification_report(y_true, y_pred)

   a) Importing the necessary libraries



```
[2] 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
```

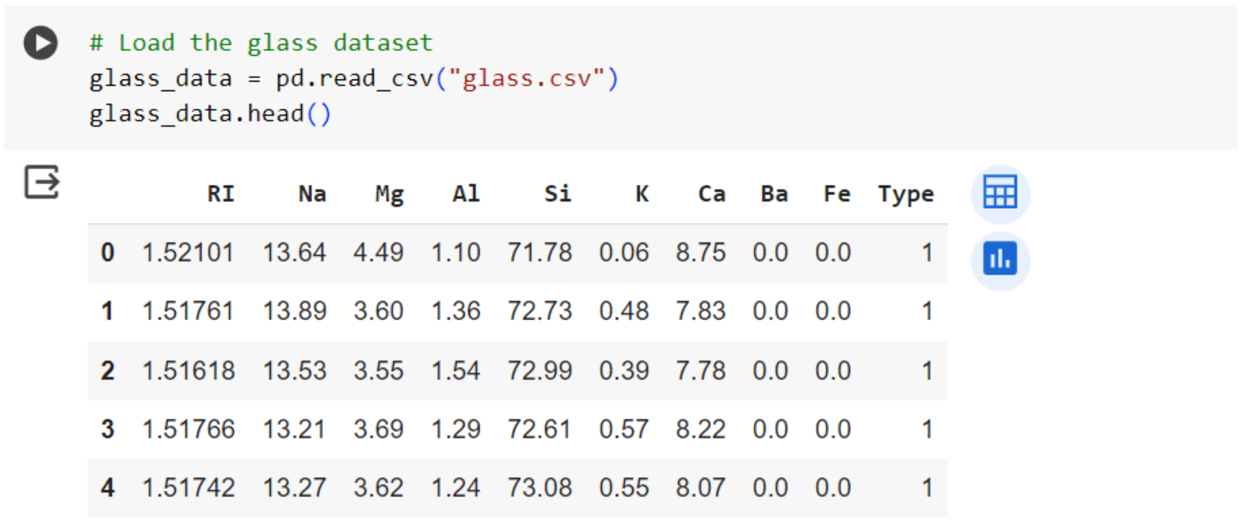
   b) Importing glass.csv file for accessing data



```
from google.colab import files
# Uploading the glass data CSV file
uploaded = files.upload()
import os
# we can see if the file uploaded in the current directory
current_directory = os.getcwd()
directory_contents = os.listdir(current_directory)
print(directory_contents)
```

```
glass.csv
• glass.csv(text/csv) - 10053 bytes, last modified: 2/8/2023 - 100% done
Saving glass.csv to glass.csv
['.config', 'glass.csv', 'sample_data']
```

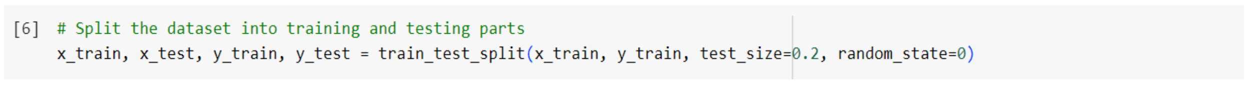
   c) Load the glass dataset



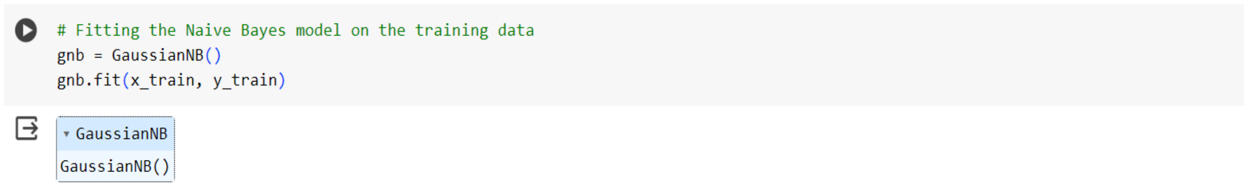
```python
# Load the glass dataset
glass_data = pd.read_csv("glass.csv")
glass_data.head()
```

	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	0.0	0.0	1
1	1.51761	13.89	3.60	1.36	72.73	0.48	7.83	0.0	0.0	1
2	1.51618	13.53	3.55	1.54	72.99	0.39	7.78	0.0	0.0	1
3	1.51766	13.21	3.69	1.29	72.61	0.57	8.22	0.0	0.0	1
4	1.51742	13.27	3.62	1.24	73.08	0.55	8.07	0.0	0.0	1

d) Split the dataset into features and target and fitting naïve bayes model on training set



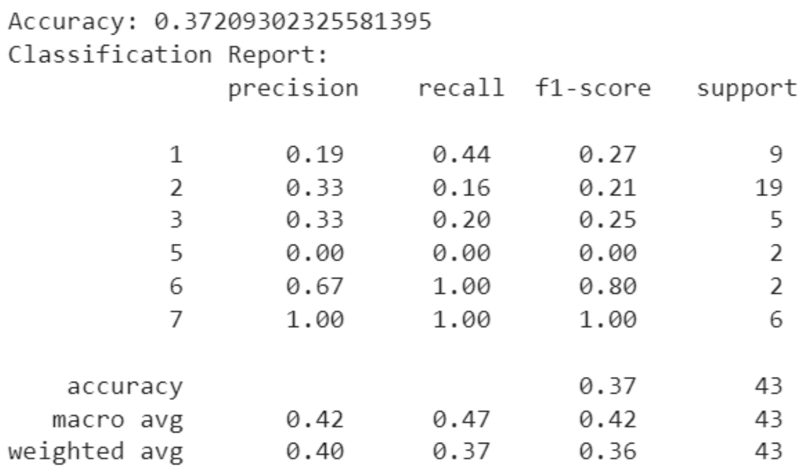
```python
[6] # Split the dataset into training and testing parts
    x_train, x_test, y_train, y_test = train_test_split(x_train, y_train, test_size=0.2, random_state=0)
```



```python
# Fitting the Naive Bayes model on the training data
gnb = GaussianNB()
gnb.fit(x_train, y_train)
```

```
▾ GaussianNB
GaussianNB()
```

e) Predicting the target on the test data and evaluating the model

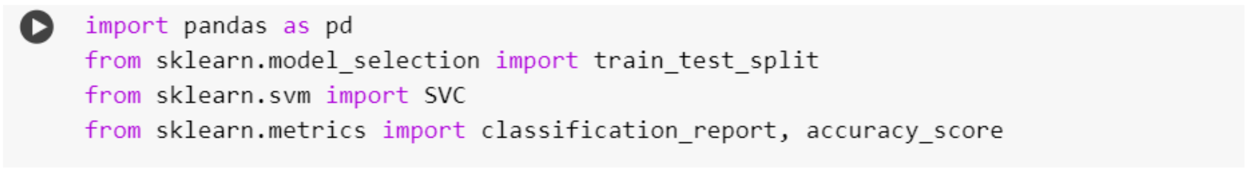


```
Accuracy: 0.37209302325581395
Classification Report:
              precision    recall  f1-score   support

           1       0.19      0.44      0.27         9
           2       0.33      0.16      0.21        19
           3       0.33      0.20      0.25         5
           5       0.00      0.00      0.00         2
           6       0.67      1.00      0.80         2
           7       1.00      1.00      1.00         6

    accuracy                           0.37        43
   macro avg       0.42      0.47      0.42        43
weighted avg       0.40      0.37      0.36        43
```

2. Implement linear SVM method using scikit library

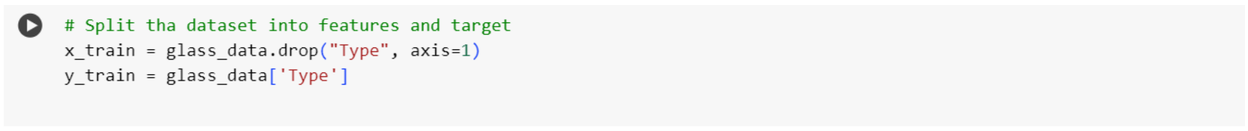
Use the same dataset above
Use train_test_split to create training and testing part
Evaluate the model on test part using score and
classification_report(y_true, y_pred)

a) Importing the necessary libraries

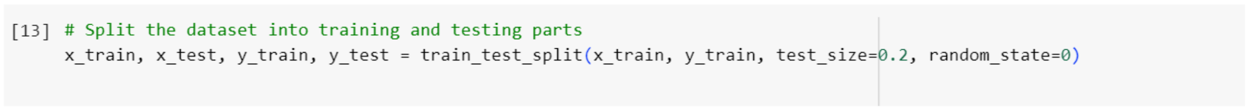


```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score
```

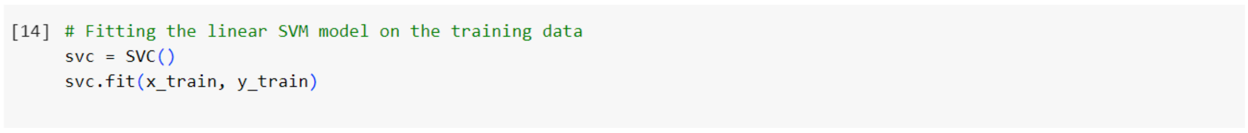
b) Split the dataset into features and target and fitting naïve bayes model on training set



```
# Split tha dataset into features and target
x_train = glass_data.drop("Type", axis=1)
y_train = glass_data['Type']
```

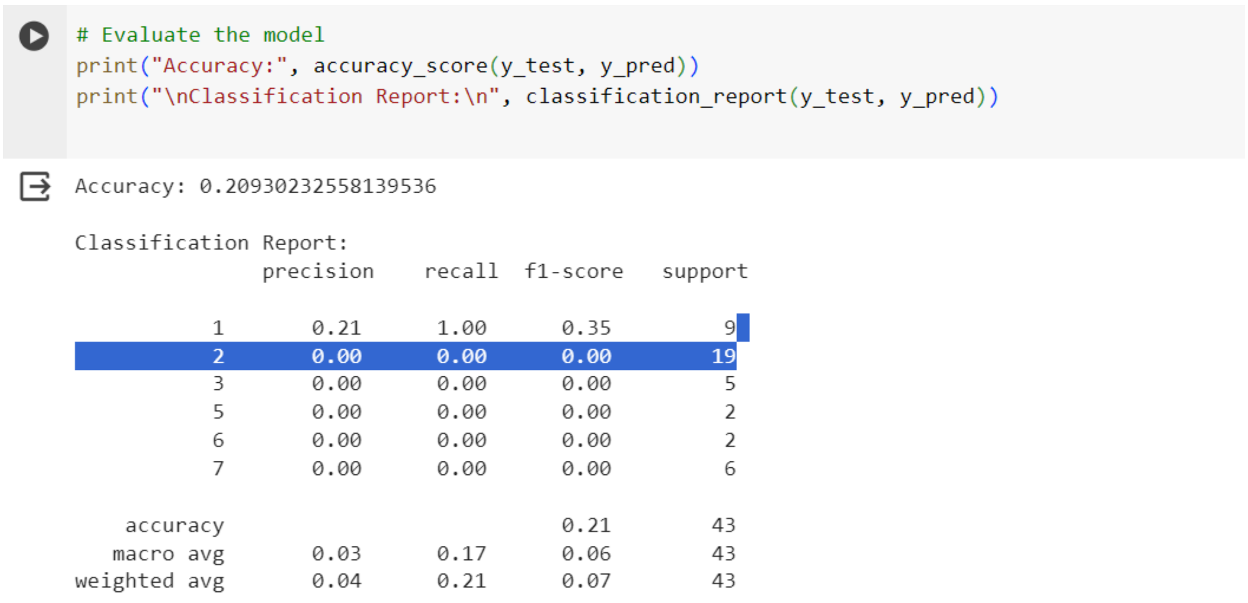


```
[13] # Split the dataset into training and testing parts
     x_train, x_test, y_train, y_test = train_test_split(x_train, y_train, test_size=0.2, random_state=0)
```



```
[14] # Fitting the linear SVM model on the training data
     svc = SVC()
     svc.fit(x_train, y_train)
```

c) Predicting the target on the test data and evaluating the model



```
# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.20930232558139536

Classification Report:
               precision    recall  f1-score   support

           1       0.21      1.00      0.35         9
           2       0.00      0.00      0.00        19
           3       0.00      0.00      0.00         5
           5       0.00      0.00      0.00         2
           6       0.00      0.00      0.00         2
           7       0.00      0.00      0.00         6

    accuracy                           0.21        43
   macro avg       0.03      0.17      0.06        43
weighted avg       0.04      0.21      0.07        43
```

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

--> We got an accuracy of 0.37209302325581395 using Naïve Bayes method and an accuracy of 0.20930232558139536 using linear SVM method. So it is evident that Naive Bayes algorithm produced better accuracy when compared to SVM method because of the maximum correct predictions. Eventhough SVM's work with both linear and non-linear data, but can be particulary useful for non-linear data