

SAI VARUN THABETI

700741122

ASSIGNMENT 5

NEURAL NETWORKS AND DEEP LEARNING

Link for the recording:

https://drive.google.com/file/d/16DKCEuP889meybNYYHW2f17n_gP00g_z/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)

```

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

glass_data = pd.read_csv('glass.csv')

x_train = glass_data.drop("Type", axis=1)
y_train = glass_data['Type']

x_train, x_test, y_train, y_test = train_test_split(x_train, y_train, test_size=0.2, random_state=0)

# Train the model using the training sets
gnb = GaussianNB()
gnb.fit(x_train, y_train)

y_pred = gnb.predict(x_test)
# Classification report
qual_report = classification_report(y_test, y_pred)
print(qual_report)
print("Naive Bayes accuracy is: ", (accuracy_score(y_test, y_pred))*100)

```

	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

Naive Bayes accuracy is: 37.2093023255814

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)

```
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

glass_data = pd.read_csv('glass.csv')

x_train = glass_data.drop("Type", axis=1)
y_train = glass_data['Type']
# splitting train and test data using train_test_split
x_train, x_test, y_train, y_test = train_test_split(x_train, y_train, test_size=0.2, random_state=0)

# Train the model using the training sets
svc = SVC()
svc.fit(x_train, y_train)
y_pred = svc.predict(x_test)
# Classification report
qual_report = classification_report(y_test, y_pred, zero_division = 0)
print(qual_report)
print("SVM accuracy is: ", accuracy_score(y_test, y_pred)*100)
```

	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

SVM accuracy is: 20.930232558139537

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

Gaussian algorithm gives better accuracy. As the accuracy we got upon training on gaussian is greater than that of SVM.