```
# Lab 8: Naive Bayes Classification Algorithm
# 0. Installation and Imports

# Uncomment if not done previously
!pip install ucimlrepo

from ucimlrepo import fetch_ucirepo
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
```

```
Requirement already satisfied: ucimlrepo in /usr/local/lib/python3.12/dist-packages (0.0.7)
Requirement already satisfied: pandas>=1.0.0 in /usr/local/lib/python3.12/dist-packages (from ucimlrepo) (2.2.2)
Requirement already satisfied: certifi>=2020.12.5 in /usr/local/lib/python3.12/dist-packages (from ucimlrepo) (2025.1
Requirement already satisfied: numpy>=1.26.0 in /usr/local/lib/python3.12/dist-packages (from pandas>=1.0.0->ucimlrep
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas>=1.0.0-
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas>=1.0.0->ucimlrepo
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas>=1.0.0->ucimlre
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->panda
```

```
# Lab 8: Naive Bayes Classification Algorithm
# 1. Load Dataset and Inspect

covertype = fetch_ucirepo(id=31)

X = covertype.data.features
y = covertype.data.targets

print("Metadata:\n", covertype.metadata)
```

```
Metadata:
 {'uci_id': 31, 'name': 'Covertype', 'repository_url': 'https://archive.ics.uci.edu/dataset/31/covertype', 'data_url'
```

```
print("Sample features:\n", X.head())
```

```
Sample features:
    Elevation  Aspect  Slope  Horizontal_Distance_To_Hydrology  \
0       2596      51      3                               258   
1       2590      56      2                               212   
2       2804     139      9                               268   
3       2785     155     18                               242   
4       2595      45      2                               153   

   Vertical_Distance_To_Hydrology  Horizontal_Distance_To_Roadways  \
0                               0                              510   
1                              -6                              390   
2                              65                             3180   
3                             118                             3090   
4                              -1                              391   

   Hillshade_9am  Hillshade_Noon  Hillshade_3pm  \
0            221             232            148   
1            220             235            151   
2            234             238            135   
3            238             238            122   
4            220             234            150   

   Horizontal_Distance_To_Fire_Points  ...  Soil_Type34  Soil_Type35  \
0                                6279  ...            0            0   
1                                6225  ...            0            0   
2                                6121  ...            0            0   
3                                6211  ...            0            0   
4                                6172  ...            0            0   

   Soil_Type36  Soil_Type37  Soil_Type38  Soil_Type39  Soil_Type40  \
0            0            0            0            0            0   
1            0            0            0            0            0   
2            0            0            0            0            0   
3            0            0            0            0            0   
4            0            0            0            0            0   

   Wilderness_Area2  Wilderness_Area3  Wilderness_Area4  
0                 0                 0                 0  
1                 0                 0                 0  
2                 0                 0                 0  
3                 0                 0                 0  
4                 0                 0                 0  
```

```
[5 rows x 54 columns]
```

```
print("Sample targets:\n", y.head())
```

```
Sample targets:
    Cover_Type
0           5
1           5
2           2
3           2
4           5
```

```
print("\nVariables:\n", covertype.variables)
```

```
54                    Wilderness_Area4  Feature  Integer        None

   description units missing_values
0         None  None             no
1         None  None             no
2         None  None             no
3         None  None             no
4         None  None             no
5         None  None             no
6         None  None             no
7         None  None             no
8         None  None             no
9         None  None             no
10        None  None             no
11        None  None             no
12        None  None             no
13        None  None             no
14        None  None             no
15        None  None             no
16        None  None             no
17        None  None             no
18        None  None             no
19        None  None             no
20        None  None             no
21        None  None             no
22        None  None             no
23        None  None             no
24        None  None             no
25        None  None             no
26        None  None             no
27        None  None             no
28        None  None             no
29        None  None             no
30        None  None             no
31        None  None             no
32        None  None             no
33        None  None             no
34        None  None             no
35        None  None             no
36        None  None             no
37        None  None             no
38        None  None             no
39        None  None             no
40        None  None             no
41        None  None             no
42        None  None             no
43        None  None             no
44        None  None             no
45        None  None             no
46        None  None             no
47        None  None             no
48        None  None             no
49        None  None             no
50        None  None             no
51        None  None             no
52        None  None             no
53        None  None             no
54        None  None             no
```

```
# Lab 8: Naive Bayes Classification Algorithm
# 2. Train-Test Split

X_train, X_test, y_train, y_test = train_test_split(
    X, y.values.ravel(), test_size=0.2, random_state=42
)

print("Training set size:", X_train.shape)
print("Testing set size:", X_test.shape)
```

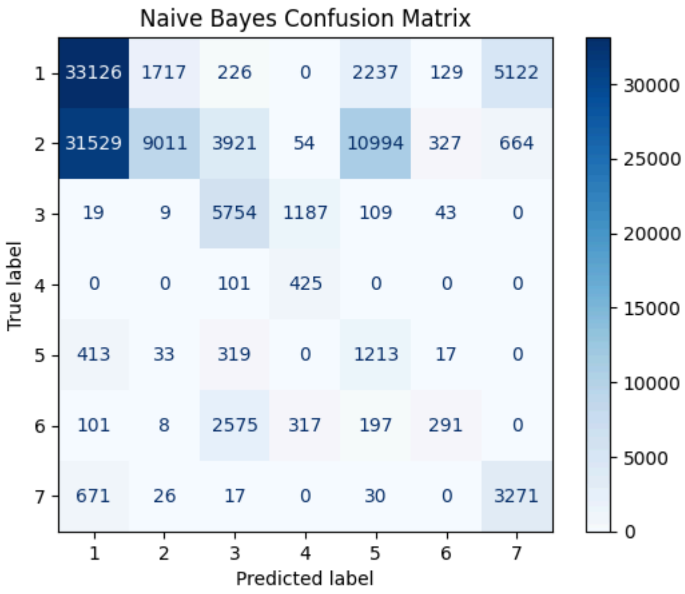
```
Training set size: (464809, 54)
Testing set size: (116203, 54)
```

```
# Lab 8: Naive Bayes Classification Algorithm
# 3. Training and Prediction

nb = GaussianNB()
nb.fit(X_train, y_train)
y_pred = nb.predict(X_test)
```

```
# Lab 8: Naive Bayes Classification Algorithm
# 4. Evaluation

print("Accuracy Score:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Accuracy Score: 0.4568814918719826
Classification Report:
               precision    recall  f1-score   support

           1       0.50      0.78      0.61     42557
           2       0.83      0.16      0.27     56500
           3       0.45      0.81      0.57      7121
           4       0.21      0.81      0.34       526
           5       0.08      0.61      0.14      1995
           6       0.36      0.08      0.14      3489
           7       0.36      0.81      0.50      4015

    accuracy                           0.46    116203
   macro avg       0.40      0.58      0.37    116203
weighted avg       0.64      0.46      0.41    116203

Confusion Matrix:
 [[33126  1717   226     0  2237   129  5122]
 [31529  9011  3921    54 10994   327   664]
 [   19     9  5754  1187   109    43     0]
 [    0     0   101   425     0     0     0]
 [  413    33   319     0  1213    17     0]
 [  101     8  2575   317   197   291     0]
 [  671    26    17     0    30     0  3271]]
```

```
# Lab 8: Naive Bayes Classification Algorithm
# 5. Confusion Matrix Visualization

from sklearn.metrics import ConfusionMatrixDisplay

cm = confusion_matrix(y_test, y_pred)
ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=nb.classes_).plot(cmap="Blues")
plt.title("Naive Bayes Confusion Matrix")
plt.show()
```



Start coding or generate with AI.