

Experiment - 5 Naive Bayes Classification

Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from sklearn.preprocessing import LabelEncoder
from sklearn.datasets import load_iris

def implement_naive_bayes(dataset_name):
    print(f"\n{' '*15} Naive Bayes on {dataset_name} Dataset {' '*15}")

    # Load the chosen dataset
    if dataset_name == "Iris":
        data = load_iris()
        X = data.data
        y = data.target
        feature_names = data.feature_names
        target_names = data.target_names
    elif dataset_name == "Wine":
        data = load_wine()
        X = data.data
        y = data.target
        feature_names = data.feature_names
        target_names = data.target_names

    # Split data into training and test sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

    # Display dataset information
    print(f"\nDataset Information:")
    print(f"Number of samples: {X.shape[0]}")
    print(f"Number of features: {X.shape[1]}")
    print(f"Number of classes: {len(target_names)}")
    print(f"Classes: {target_names}")
    print(f"Features: {feature_names}")
    print(f"\nTraining set size: {X_train.shape[0]}")
    print(f"Test set size: {X_test.shape[0]}")

    # Initialize and train the Naive Bayes classifier
    print("\nTraining Naive Bayes classifier...")
    naive_bayes = GaussianNB()
    naive_bayes.fit(X_train, y_train)
```

```

# Make predictions
y_pred = naive_bayes.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"\nAccuracy: {accuracy:.4f} ({accuracy*100:.2f}%)")

# Display confusion matrix
cm = confusion_matrix(y_test, y_pred)
print("\nConfusion Matrix:")
print(cm)

# Display classification report
print("\nClassification Report:")
report = classification_report(y_test, y_pred, target_names=target_names)
print(report)

# Visualize confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=target_names,
            yticklabels=target_names)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title(f'Confusion Matrix - {dataset_name} Dataset')
plt.tight_layout()
plt.savefig(f"naive_bayes_{dataset_name.lower()}_confusion_matrix.png")

# Visualize probability distributions
# For simplicity, we'll just plot the first two features for each class
plt.figure(figsize=(12, 6))

# Create a DataFrame for easier plotting
df = pd.DataFrame(X, columns=feature_names)
df['target'] = y
df['target_name'] = [target_names[i] for i in y]

# Plot the first two features
plt.subplot(1, 2, 1)
sns.scatterplot(x=feature_names[0], y=feature_names[1], hue='target_name',
data=df)
plt.title(f'{feature_names[0]} vs {feature_names[1]}')

# Plot another set of features if available
if len(feature_names) > 3:
    plt.subplot(1, 2, 2)
    sns.scatterplot(x=feature_names[2], y=feature_names[3],
hue='target_name', data=df)
    plt.title(f'{feature_names[2]} vs {feature_names[3]}')

plt.tight_layout()
plt.savefig(f"naive_bayes_{dataset_name.lower()}_feature_distribution.png")

```

```

        return accuracy, report

def main():
    # Implement Naive Bayes on Iris dataset
    implement_naive_bayes("Iris")

    # Show the plots
    plt.show()

if __name__ == "__main__":
    main()

```

Output

```
PS V:\Deeptanshu Lal\PROJECTS\ML> python .\exp-6\exp-6.py
```

```
===== Naive Bayes on Iris Dataset =====
```

Dataset Information:

Number of samples: 150

Number of features: 4

Number of classes: 3

Classes: ['setosa' 'versicolor' 'virginica']

Features: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']

Training set size: 105

Test set size: 45

Training Naive Bayes classifier...

Accuracy: 0.9778 (97.78%)

Confusion Matrix:

```
[[19  0  0]
 [ 0 12  1]
 [ 0  0 13]]
```

Classification Report:

	precision	recall	f1-score	support
setosa	1.00	1.00	1.00	19
versicolor	1.00	0.92	0.96	13
virginica	0.93	1.00	0.96	13
accuracy			0.98	45
macro avg	0.98	0.97	0.97	45

weighted avg	0.98	0.98	0.98	45
--------------	------	------	------	----

Visual Outputs

