

**Name : Ch Mubashir**

SAP : 56892.

Course : Data Mining .

## Lab Task 07

## Lab: k-NN vs Naïve Bayes (Clean + Noisy)

## **Objectives:**

1. Implement the k-Nearest Neighbors (k-NN) classifier.
  2. Implement the Naïve Bayes classifier.
  3. Compare the performance of the two algorithms.

## Tools: Scikit-learn

Dataset: Wine Dataset (from `sklearn.datasets import load_wine`)

## Lab Tasks:

1. Load the Wine dataset and split it into training and testing sets.

#### 1. Load and Split the Dataset

```
] 0s
1 # Load the Wine dataset
2 wine = load_wine()
3 X = wine.data
4 y = wine.target
5
6 # Split into training (80%) and testing (20%) sets
7 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
8
```

#### 2. Standardize the features (important for k-NN!).

##### 2. Standardize the Features

```
] 
1 scaler = StandardScaler()
2 X_train_scaled = scaler.fit_transform(X_train)
3 X_test_scaled = scaler.transform(X_test)
```

#### 3. Train a k-NN classifier (KNeighborsClassifier) with n\_neighbors=5. Calculate its accuracy.

##### 3. Train k-NN Classifier

```
] 0s
1 # k-NN with k=5
2 knn = KNeighborsClassifier(n_neighbors=5)
3 knn.fit(X_train_scaled, y_train)
4 y_pred_knn = knn.predict(X_test_scaled)
5
6 # Accuracy
7 acc_knn = accuracy_score(y_test, y_pred_knn)
8 print(f"k-NN Accuracy (k=5): {acc_knn:.3f}")
9
```

→ k-NN Accuracy (k=5): 0.972

#### 4. Train a Naïve Bayes classifier (GaussianNB). Calculate its accuracy.

Tools Help

In all ▾

X 4. Train Naïve Bayes Classifier

```
[10] ✓ 0s
1 nb = GaussianNB()
2 nb.fit(X_train, y_train) # NB does not require scaling
3 y_pred_nb = nb.predict(X_test)
4
5 acc_nb = accuracy_score(y_test, y_pred_nb)
6 print(f"Naïve Bayes Accuracy: {acc_nb:.3f}")

→ Naïve Bayes Accuracy: 0.972
```

5. Compare the accuracy of the two models.

5. Compare the Two Models

```
11] ✓ 0s
1 print("\n--- Model Comparison ---")
2 print(f"k-NN (k=5) Accuracy: {acc_knn:.3f}")
3 print(f"Naïve Bayes Accuracy: {acc_nb:.3f}")

→
--- Model Comparison ---
k-NN (k=5) Accuracy: 0.972
Naïve Bayes Accuracy: 0.972
```

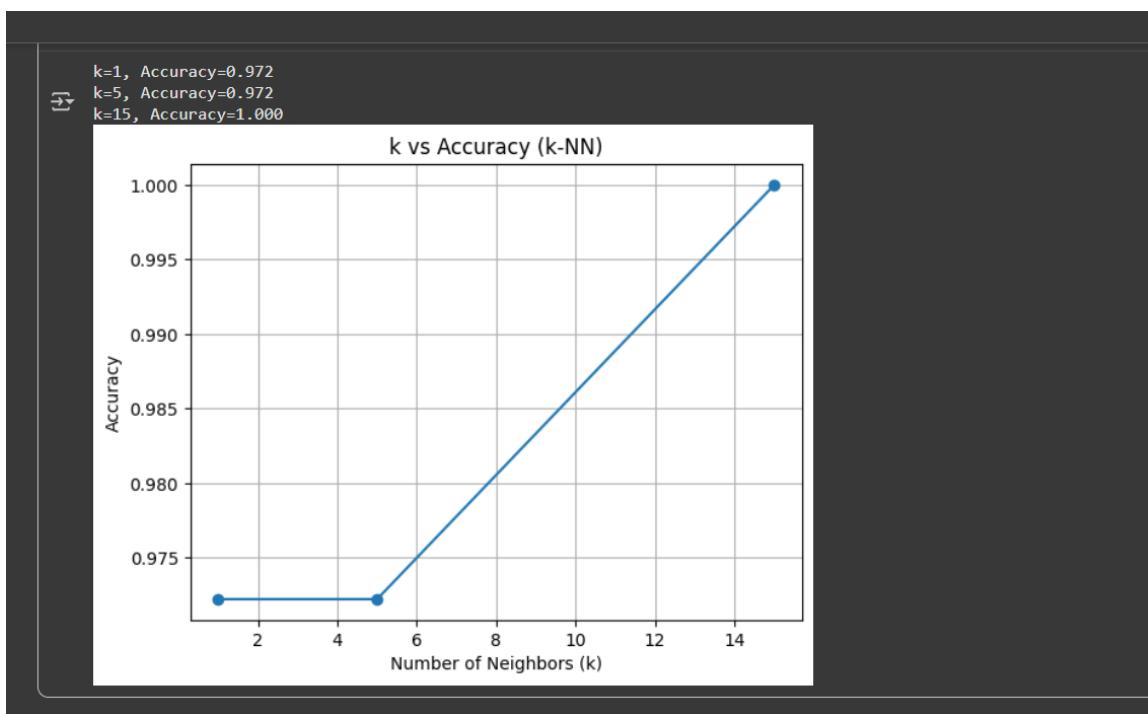
Excercise :

1. For k-NN, experiment with different values of k (1, 5, 15). Plot k against accuracy. What is the optimal k and why does performance change?

1. k-NN for Different k Values

```
[12]
1 k_values = [1, 5, 15]
2 accuracies = []
3 for k in k_values:
4     knn = KNeighborsClassifier(n_neighbors=k)
5     knn.fit(X_train_scaled, y_train)
6     y_pred = knn.predict(X_test_scaled)
7     acc = accuracy_score(y_test, y_pred)
8     accuracies.append(acc)
9     print(f"k={k}, Accuracy={acc:.3f}")
10
11 plt.plot(k_values, accuracies, marker='o')
12 plt.title("k vs Accuracy (k-NN)")
13 plt.xlabel("Number of Neighbors (k)")
14 plt.ylabel("Accuracy")
15 plt.grid(True)
16 plt.show()

→ k=1, Accuracy=0.972
→ k=5, Accuracy=0.972
→ k=15, Accuracy=1.000
```



## Effect of Different k Values in k-NN

- Experiment with different values of **k** in the k-Nearest Neighbors algorithm (e.g.,  $k = 1, 5, 15$ ).
- Plot the accuracy of the model against the value of **k**.
- Identify which value of **k** gives the best performance and explain **why the accuracy changes** as **k** increases.

2. Why is feature scaling critical for k-NN but not for Naïve Bayes? Test this by running k-NN on the unscaled data and reporting the accuracy.

```
2. Test k-NN Without Feature Scaling

1 # k-NN on unscaled data
2 knn_unscaled = KNeighborsClassifier(n_neighbors=5)
3 knn_unscaled.fit(X_train, y_train)
4 y_pred_unscaled = knn_unscaled.predict(X_test)
5
6 acc_unscaled = accuracy_score(y_test, y_pred_unscaled)
7 print(f"k-NN Accuracy without scaling: {acc_unscaled:.3f}")
8
```

→ k-NN Accuracy without scaling: 0.806

### Importance of Feature Scaling

- Explain **why feature scaling is critical for the k-NN algorithm** but not for the Naïve Bayes classifier.
- Test this concept by running k-NN on **unscaled data** and observe how the accuracy changes compared to when the data is scaled.
- Discuss the reason for this difference based on how each algorithm works.