

**Project Report: Handwritten Digit Classification and Clustering**

**Submitted To:** Ma’am Namra Sheikh

**Subject:** Artificial Intelligence (Lab)

**Group Members**

**Zunaira Kabeer 2022-CS-702**

**Aimen Munir 2022-CS-703**

Contents

[1](#_Toc185181310)

[**1. Project Overview** 2](#_Toc185181311)

[**2. Objectives** 2](#_Toc185181312)

[**3. Dataset Details** 2](#_Toc185181313)

[**About the data set:** 2](#_Toc185181314)

[**4. Data Preprocessing** 2](#_Toc185181315)

[**5. Supervised Learning: Naive Bayes Classifier** 2](#_Toc185181316)

[**5.4 Analysis on Classification Report** 2](#_Toc185181317)

[**4. Key Observations** 2](#_Toc185181318)

[**Key Observations:** 2](#_Toc185181319)

[**Accuracy and Insights:** 2](#_Toc185181320)

[**6. Unsupervised Learning: K-Means Clustering** 2](#_Toc185181321)

[Key Observations of Clustering : 2](#_Toc185181322)

[Insights on Clustering Performance: 2](#_Toc185181323)

[**7. Key Insights and Challenges** 2](#_Toc185181324)

[**8. Conclusion** 2](#_Toc185181325)

**1. Project Overview**

This project focuses on recognizing handwritten digits from a dataset using **Naive Bayes Classification** for supervised learning and **K-Means Clustering** for unsupervised learning. The dataset comprises numerical values representing images of handwritten digits (0-9), a commonly used dataset for benchmarking machine learning models.

**2. Objectives**

1. Develop a **Naive Bayes Classification model** to classify handwritten digits.
2. Use **Principal Component Analysis (PCA)** for dimensionality reduction and visualize the clustering structure.
3. Apply the **K-Means Clustering Algorithm** to explore unsupervised learning and determine the optimal number of clusters using the **Elbow Method**.
4. Evaluate model performance through metrics like Accuracy, Precision, Recall, F1-score, and Confusion Matrix.
5. Visualize the results of the model through confusion matrix and cluster visualization

**3. Dataset Details**

**About the data set:**

1. This data set has a total of **5.6K instances.**
2. There are **64 features** each of which are **continuous numeric data.**
3. There are a total of **10 output classes for each digit from 0 to 9**. The 64 features are extracted from 32x32 bitmaps which are divided into non-overlapping blocks of 4x4 and the number of on pixels are counted in each block. This generates an input matrix of 8x8 (64 values) where each element is an integer in the range 0-16. This reduces dimensionality and gives invariance to small distortions.
4. **Source**:https://archive.ics.uci.edu/dataset/80/optical+recognition+of+handwritten+digits).
5. **Format**: Each row represents an image, where pixel values are stored as numerical features.

**4. Data Preprocessing**

* **Data Cleaning**: Checked for null/missing values and removed redundant data.
* **Dimensionality Reduction**: Principal Component Analysis (PCA) reduced high-dimensional pixel data to 2 components for visualization purposes.
* **Feature Scaling**: Scaled data to ensure all features are on the same scale for effective clustering and classification.
* **Balanced Classes**: Checked if all our output classes has balanced instances

**5. Supervised Learning: Naive Bayes Classifier**

**5.1 Model Description**

Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem. Given the dataset, it assumes features are conditionally independent.

**5.2 Results**

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.80 (80.16%) |
| Precision | 0.86 |
| Recall | 0.80 |
| F1-Score | 0.81 |
|  |  |

**5.3 Classification Report**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Class (Digit)* | *Precision* | *Recall* | *F1-Score* | *Support* |
| 0 | 0.92 | 1.00 | 0.96 | 108 |
| 1 | 0.79 | 0.84 | 0.82 | 102 |
| 2 | 0.99 | 0.70 | 0.82 | 107 |
| 3 | 0.96 | 0.61 | 0.75 | 118 |
| 4 | 0.99 | 0.64 | 0.78 | 117 |
| 5 | 0.97 | 0.67 | 0.79 | 97 |
| 6 | 0.96 | 0.97 | 0.96 | 123 |
| 7 | 0.74 | 0.98 | 0.84 | 124 |
| 8 | 0.45 | 0.95 | 0.61 | 105 |
| 9 | 0.85 | 0.64 | 0.73 | 123 |

* **Accuracy**: 80.16% of digits were classified correctly.
* **Observations**:
  + Digits like **0** and **6** showed high accuracy due to their clear shape.
  + Confusion occurred for similar-looking digits such as **3 and 8** or **4 and 9**.

**5.4 Analysis on Classification Report**

**1. Overall Performance**

* The model achieved varying performance across different classes, with **Precision**, **Recall**, and **F1-Score** showing a wide range.
* Classes such as **0, 6, and 7** performed particularly well, while others like **3, 4, 8**, and **9** struggled with either precision, recall, or both.

**2. High-Performing Classes**

* **Digit 0**: Achieved perfect **Recall (1.00)** and a very high **Precision (0.92)**, leading to an **F1-Score of 0.96**. This suggests the model easily identifies '0'.
* **Digit 6**: Maintains **high Precision (0.96)** and **Recall (0.97)**, giving it an **F1-Score of 0.96**. '6' has clear distinguishing features.
* **Digit 7**: Despite moderate **Precision (0.74)**, it achieves very high **Recall (0.98)**, resulting in a good **F1-Score (0.84)**. This indicates that while '7' is often correctly identified, it is sometimes mistaken for other digits.

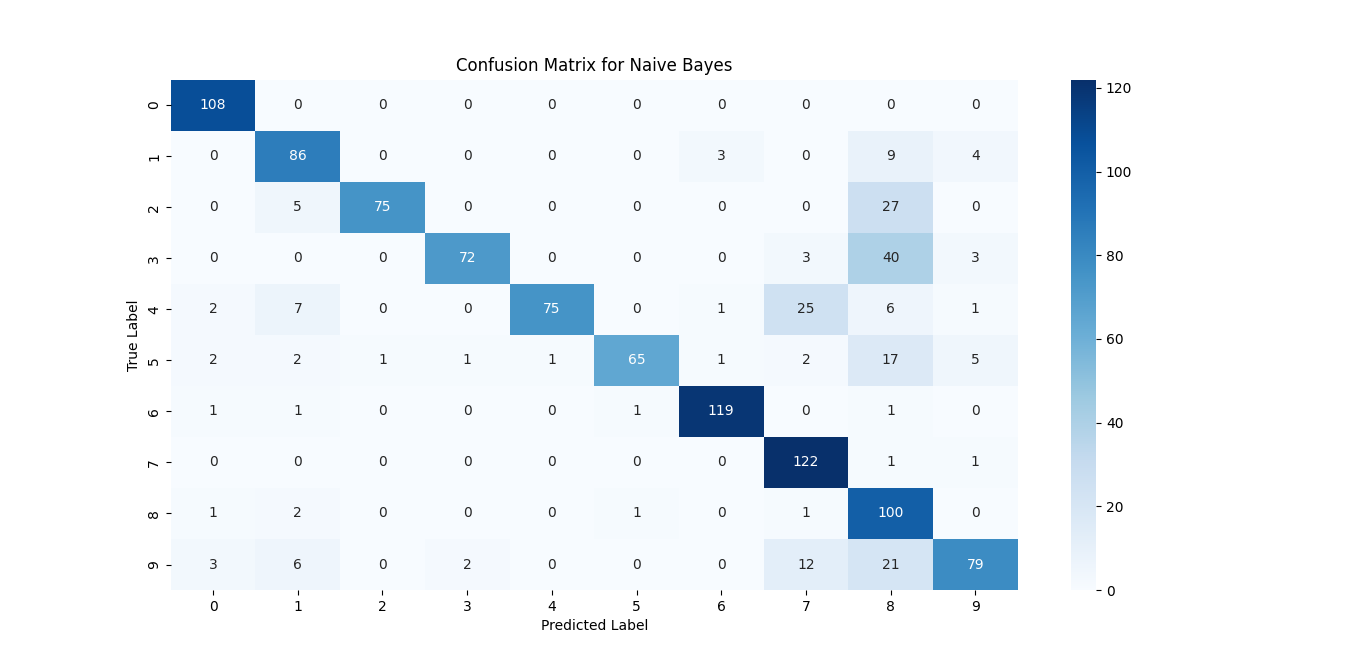
**3. Challenging Classes**

* **Digit 8**: The worst-performing class with **Precision (0.45)** and **Recall (0.95)**.
  + **Analysis**: The high recall indicates the model correctly identifies most '8's, but the very low precision shows a high false positive rate. This means the model frequently misclassifies other digits as '8'.
  + **Reason**: Digit '8' has a shape that resembles digits like '3' and '0', leading to ambiguity.
* **Digit 3**: Has a strong **Precision (0.96)** but a weak **Recall (0.61)**, resulting in an **F1-Score of 0.75**.
  + **Analysis**: The low recall suggests many true '3's are missed and classified as other digits (e.g., '8' or '5').
* **Digit 4** and **Digit 9**: Both share similar trends of high precision but low recall:
  + **Digit 4**: Precision = **0.99**, Recall = **0.64** → Model is overconfident about the predictions but misses true '4's.
  + **Digit 9**: Precision = **0.85**, Recall = **0.64** → Similar misclassification patterns.

**4. Key Observations**

* **Misclassification Trends**:
  + Digits with similar shapes (e.g., 3, 8, 5 or 4, 9) tend to confuse the model.
  + High recall but low precision (like for '8') suggests frequent misclassification of other digits as '8'.
* **Performance Variation**:
  + Digits like '0' and '6' are easier to classify due to distinct features.
  + Digits like '3' and '8' struggle due to overlapping features.

**5.5 Confusion Matrix (Visualization)**

The confusion matrix represents the performance of the **Naive Bayes classifier** on a dataset of handwritten digits (0–9). Below is a detailed analysis of the confusion matrix:

**Key Observations:**

1. **Diagonal Dominance**:
   * The diagonal entries represent correctly predicted digits (True Positives).
   * The matrix shows that digits **6, 7, and 0** have relatively high counts on the diagonal:
     + **Digit 6**: 119 correctly classified.
     + **Digit 7**: 122 correctly classified.
     + **Digit 0**: 108 correctly classified.
   * These digits are relatively easier for the model to recognize, possibly due to distinct features.
2. **Misclassifications**:
   * Misclassifications occur where non-diagonal entries are observed (e.g., row 1, column 8).
   * Some significant confusions:
     + **Digit 3** is confused with **8** (40 predictions).
     + **Digit 2** is confused with **8** (27 predictions).
     + **Digit 9** is confused with **8** (21 predictions).
   * This indicates that digits **2, 3, 8, and 9** share similar structural features, leading to confusion.
3. **Worst Performing Digits**:
   * **Digit 8**: Frequently misclassified as **3** (21) and as **9** (21).
   * **Digit 5**: Only **65** correctly classified out of its total count; misclassifications occur across many classes.
4. **Overall Performance**:
   * The classifier performs well for certain digits but struggles with others that have overlapping or ambiguous features.
   * Digits **8** and **5** appear to be the most challenging for the model.

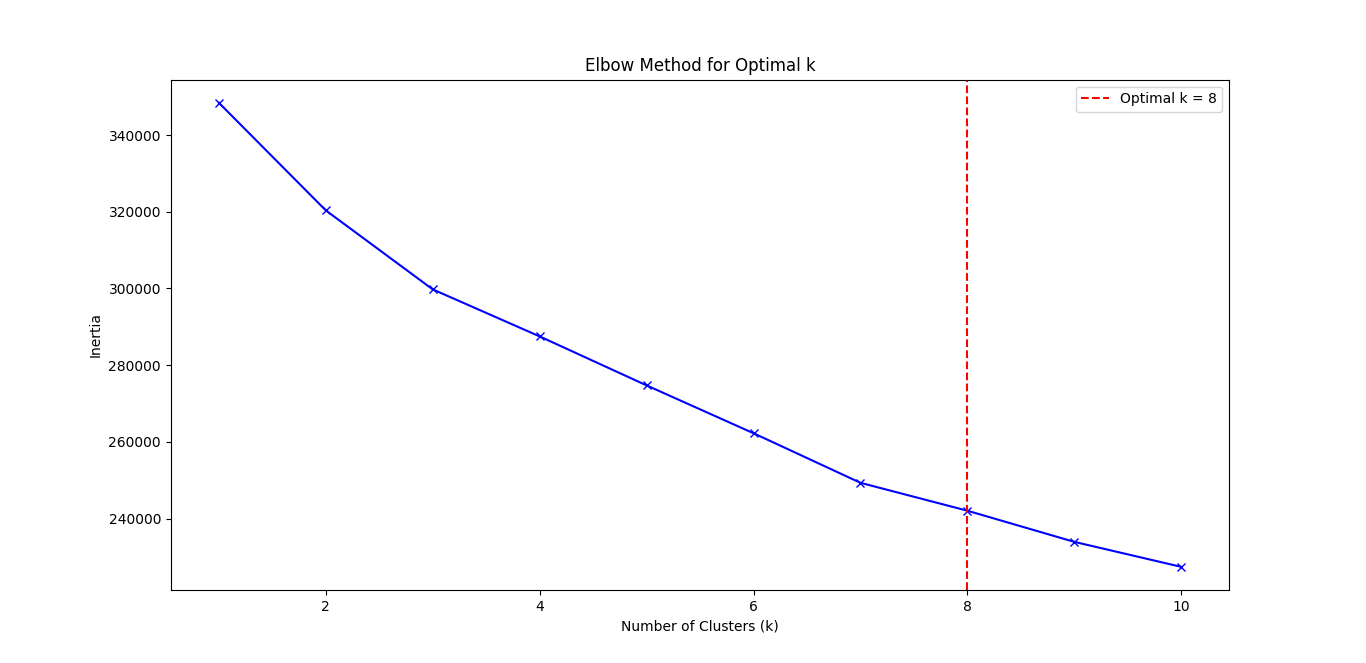
**Accuracy and Insights:**

* **True Positives** are high for digits like **6, 7, and 0**.
* **False Positives** and **False Negatives** are prominent for **8, 5**, and **3**, leading to lower performance for these classes.
* **Reasons for Confusion**:
  + Similarity in shape: For example, **3 and 8** share curves.
  + Overlapping pixel features in the data representation.

**6. Unsupervised Learning: K-Means Clustering**

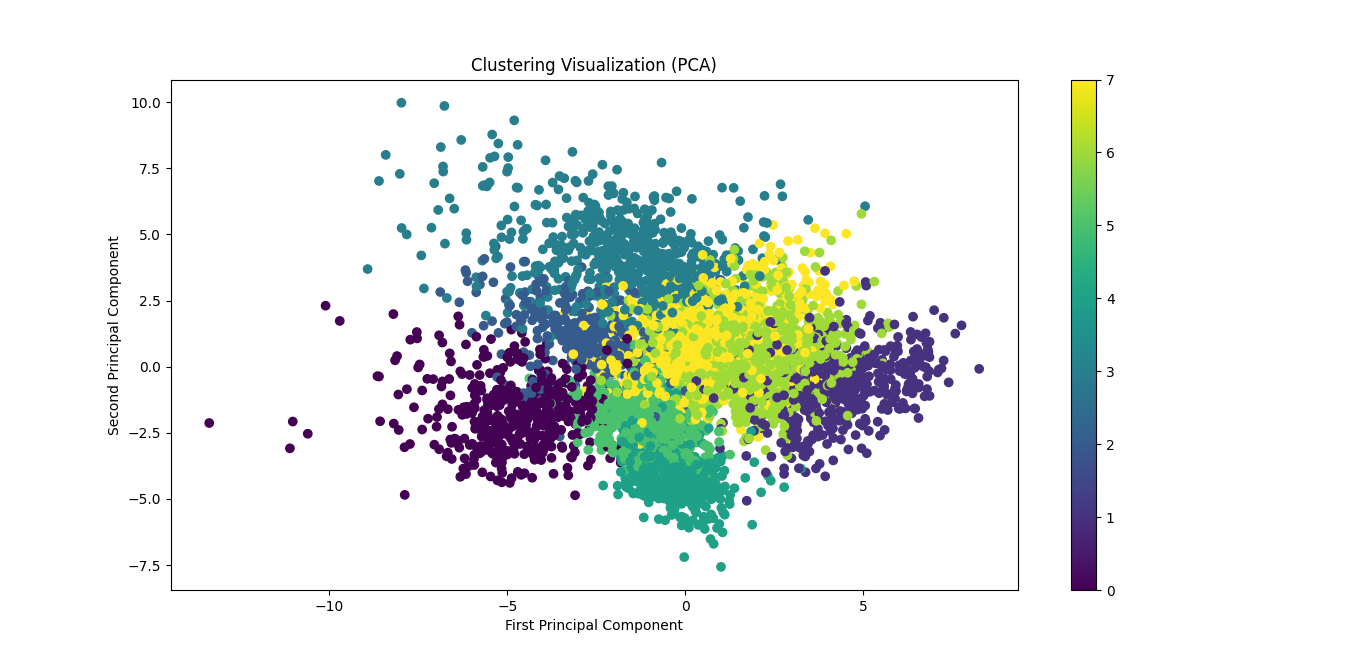
**6.1 Optimal Number of Clusters**

The **Elbow Method** determined the optimal number of clusters (k) as **8**.

* The elbow point occurs at k = 8, where the inertia starts decreasing less sharply.
* While there are 10 true classes (digits), the clustering algorithm identifies 8 groups, likely because:
  + Some digits have overlapping shapes (e.g., **3 and 8** or **4 and 9**).
  + Clustering works on numerical patterns, not true labels.

**6.2 Visualization using PCA**

Principal Component Analysis (PCA) projected the data onto 2D space for clustering visualization.

* Points with similar features form clusters.
* Clusters overlap for similar-looking digits, reflecting inherent ambiguities in the dataset.

### ****Key Observations of Clustering****:

1. **Dimensionality Reduction**:
   * The original high-dimensional data is projected into **2D space** using PCA.
   * The axes, "First Principal Component" and "Second Principal Component," capture the maximum variance in the data.
2. **Cluster Separation**:
   * The data points are color-coded to represent different clusters (ranging from 0 to 7, based on the color bar).
   * There is **partial overlap** between the clusters, particularly around the center, indicating difficulty in distinguishing certain digits.
3. **Tightly Packed Regions**:
   * Some clusters, especially those in the **lower-left region** (e.g., represented by darker colors like 0), are tightly packed and well-separated.
   * This suggests that some digits (e.g., **0**) are easily distinguishable due to distinct features.
4. **Overlapping Clusters**:
   * Clusters in the **middle-right** region (e.g., represented by lighter shades like 5, 6, or 7) overlap significantly.
   * This overlap indicates that certain digits share similar pixel patterns, leading to challenges in clustering (e.g., **5, 6, and 7** often share loops or curves).
5. **Outliers**:
   * A few data points are spread farther away from the central clusters.
   * These may represent **ambiguous digits** or **misclassified samples** where the handwriting style does not conform to typical patterns.

### ****Insights on Clustering Performance****:

1. **Cluster Quality**:
   * Clusters for digits like **0 and 3** appear more compact and separable, indicating good clustering performance for these digits.
   * In contrast, digits like **5, 6, and 7** have dispersed clusters and overlap, suggesting poor distinction in their feature space.
2. **Variability in Digits**:
   * The overlapping nature of certain clusters implies that digits in the dataset exhibit significant **intra-class variability**.
   * For instance, a "7" written with a horizontal line might resemble a "1," or a poorly closed "6" might resemble an "8."

**6.3 Clustering Accuracy**

* **Clustering Accuracy**: 78.13%  
  The clustering algorithm effectively grouped similar digits but fell short due to:
* Overlap between certain classes.
* Dimensionality reduction limiting feature separation.

**7. Key Insights and Challenges**

1. **Supervised Learning**:
   * The **Naive Bayes model** achieved good performance, especially for distinct digits like **0, 6**, and **7**.
   * Misclassifications highlight the need for feature engineering or using more complex models (e.g., SVM or Neural Networks).
2. **Unsupervised Learning**:
   * The **K-Means algorithm** identified 8 clusters instead of 10, revealing overlaps in digit representation.
   * PCA provided effective visualization but may lose information during dimensionality reduction.
3. **Challenges**:
   * Ambiguous shapes between digits caused errors.
   * Dataset noise or pixel variations impacted clustering and classification performance.

**8. Conclusion**

The project successfully demonstrated both supervised and unsupervised machine learning techniques for handwritten digit recognition:

* **Naive Bayes Classification** achieved **80.16% accuracy**.
* **K-Means Clustering** identified patterns with an accuracy of **78.13%**, with **k=8** clusters as optimal.