Handwritten Digit Recognition Using KNN and Naive Bayes

Muddasar Razzaq
Information Tech Consultants

A comparison of KNN and Naive Bayes classifiers on the MNIST dataset.

Objective and Dataset

Objective: Build a model to recognize handwritten digits.

Dataset: MNIST (60,000 training, 10,000 testing images).
Each image is 28x28 pixels converted to 784 features.

Steps: Data cleaning, normalization, and splitting.

KNN Implementation

Concept: Classifies samples using k nearest neighbors.

Distance: Euclidean.

Results:
- Accuracy: ~80%
- Runtime: ~60 sec

Pros: High accuracy.
Cons: Computationally expensive.

Naive Bayes Implementation

Concept: Probabilistic model assuming independent features.

Formula: $P(y|X) \propto P(y) \prod P(xi|y)$

Results:
- Accuracy: ~60%
- Runtime: ~2 sec

Pros: Very fast.
Cons: Lower accuracy.

Comparison Table

Model	Accuracy	Runtime	Notes
KNN	80%	~60 sec	More accurate, slower
Naive Bayes	60%	~2 sec	Fast, less accurate

Confusion Matrices

KNN Confusion Matrix (strong diagonal)
Naive Bayes Confusion Matrix (more spread)

Diagonal = correct predictions
Off-diagonal = misclassifications

Key Insights

• KNN: High accuracy, slower.
• Naive Bayes: Fast, less accurate.
• Both useful for learning classic ML.

Takeaway: KNN performs better overall on MNIST.

Conclusion

- Both models implemented without scikit-learn.
- Demonstrates power of normalization and clean data.
- Future work: explore neural networks for better accuracy.

Acknowledgments

Project by: Muddasar Razzaq
Organization: Information Tech Consultants
Tools: Python, NumPy, Pandas, Matplotlib, Google Colab
Dataset: MNIST