Author(s): Pinkesh Tandel and Dennis Darko

Institution: Northeastern University
College: College of Professional Studies
Location: Vancouver, BC, Canada
Course: ALY6020: Predictive Analytics

Date: October 20, 2024

Under the guidance of Dr. Nina Rajabi Nasab

**Title: Handwriting Recognition Using K-Nearest Neighbors and Neural Networks**

**Abstract**
This study aims to predict handwritten digits using machine learning models, particularly K-Nearest Neighbors (KNN) and Neural Networks. Through dimensionality reduction techniques like Principal Component Analysis (PCA), we reduce the complexity of the dataset. After training and testing both models, we compare their performance based on accuracy and other classification metrics, such as precision, recall, and F1-score. This comparison provides insight into which model performs better and how they can be used for handwriting recognition to support the school in identifying students who may need assistance in improving motor skills.

## Introduction

In early education, motor skills play a vital role in a child's development. To help schools identify children who may need assistance with their motor skills, we designed a handwriting recognition system. The system leverages machine learning to predict what digits students have drawn from their handwriting. Two models are tested for this purpose: a simple K-Nearest Neighbors (KNN) algorithm and a more advanced Neural Network (NN). This study applies dimensionality reduction techniques such as Principal Component Analysis (PCA) to simplify the dataset and improve model efficiency.

## Part 1: KNN Model for Handwriting Recognition

K-Nearest Neighbors (KNN) is a simple yet powerful algorithm used for classification. It works by finding the 'k' nearest data points to the test data and classifying based on majority voting. In this project, we applied PCA to reduce the dimensionality of the feature space before training the KNN model.

- **Accuracy**: The KNN model achieved an accuracy of **64%** on the test set.
- **Classification Metrics**:

    - Precision, Recall, and F1-score for each digit are reported.
    - For instance, digit '0' had a precision of 0.81 and recall of 0.88, while digit '9' showed lower performance with a precision of 0.46 and recall of 0.39.
- **Challenges**:
    - KNN is computationally expensive for large datasets since it calculates the distance between the test point and all training points.
    - Dimensionality reduction (PCA) helped speed up the process, but the model struggled with recognizing more complex digits, such as 7, 8, and 9.
- The performance of KNN is acceptable for simpler digits but declines for digits with more complex features. Despite its simplicity, KNN might not be the best choice for handwriting recognition with a large feature set.

## Part 2: Neural Network Model for Handwriting Recognition

Neural Networks are more complex models that can capture non-linear relationships in the data. We implemented a basic feedforward Neural Network with two hidden layers using PCA-transformed features.

- **Accuracy**: The Neural Network achieved a higher accuracy of **69%** on the test set, outperforming KNN.
- **Classification Metrics**:
    - The Neural Network showed better performance across all digits. For example, digit '0' had a precision of 0.87 and recall of 0.87, while digit '9' improved to a precision of 0.47 and recall of 0.49.
    - The precision and recall for digits like 2, 3, and 4 showed significant improvement compared to the KNN model.
- **Challenges**:
    - Neural Networks require significant computational power and tuning. Although the model performed better, training time was longer.
    - Overfitting is a concern, but this was mitigated by using PCA to reduce dimensionality.
- The Neural Network's ability to recognize more complex patterns in the data allowed it to outperform the KNN model, but it also comes with the tradeoff of increased computational cost and longer training times.

## Part 3: Comparison and Recommendation

Using multiple benchmarking metrics, we compared the performance of KNN and Neural Networks. While both models were tested using the same PCA-transformed dataset, the Neural Network consistently outperformed KNN in terms of accuracy and precision.

- **KNN vs Neural Network**:
  - **Accuracy**: The Neural Network achieved an accuracy of 69%, compared to 64% for KNN.
  - **Precision, Recall, F1-score**: Across all classes, the Neural Network performed better, especially for more complex digits like 7, 8, and 9, where KNN struggled.
- **Recommendation**:
  - Given the results, the Neural Network is the preferred model for this task. Its ability to recognize patterns and handle more complex digits makes it a better choice for the school to use in identifying students with motor skill challenges.
  - However, if computational efficiency and simplicity are the primary concerns, KNN could still be considered for smaller datasets or simpler tasks.

## Conclusion

In this project, we explored two models for handwriting recognition: KNN and Neural Networks. By applying PCA to reduce dimensionality, we improved the performance of both models. The Neural Network model showed superior performance and is recommended for use in the school's handwriting recognition tasks. Future work may include tuning the Neural Network further or exploring other classification algorithms such as Random Forests or Support Vector Machines (SVM).

## References


Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). **Scikit-learn: Machine Learning in Python**. *Journal of Machine Learning Research*, *12*, 2825–2830. Retrieved from https://jmlr.org/papers/v12/pedregosa11a.html

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., & Zheng, X. (2016). **TensorFlow: A System for Large-Scale Machine Learning**. *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 265–283. Retrieved from https://www.tensorflow.org

Waskom, M. L. (2021). **Seaborn: Statistical Data Visualization**. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021

Van Der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). **The NumPy Array: A Structure for Efficient Numerical Computation**. *Computing in Science & Engineering*, *13*(2), 22-30. https://doi.org/10.1109/MCSE.2011.37

McKinney, W. (2010). **Data Structures for Statistical Computing in Python**. *Proceedings of the 9th Python in Science Conference*, 56-61. https://doi.org/10.25080/Majora-92bf1922-00a

Kingma, D. P., & Ba, J. (2015). **Adam: A Method for Stochastic Optimization**. *International Conference on Learning Representations*. Retrieved from https://arxiv.org/abs/1412.6980

## Appendix