*OSMAN, ABDULLAHI OMAR*

*Cloud Computing and Big Data Program*

*Seneca Polytechnic*

*Handwritten Digit Classification*

Machine Learning Project Report

# Report on Handwritten Digit Classification Project

## Project Overview

The project focuses on classifying handwritten digits (0-9) using the **MNIST dataset**, which consists of 70,000 grayscale images of handwritten digits. The dataset is divided into 60,000 training images and 10,000 test images, each of size 28x28 pixels. The goal is to build a machine learning model that can accurately classify these images into their respective digit categories.

## Data Preprocessing

### Data Loading:

* The MNIST dataset was loaded using the ‘fetch\_openml’ library from ‘scikit-learn’.
* The ‘Image’ library from ‘PIL’ was used to save images of each digit into corresponding folders on the disk.

### Normalization:

* Pixel values were normalized to the range [0, 1] by dividing by 255.0 and converting to the float data type.

### Reshaping:

* + Labels were converted to one-hot encoding for the **Feed-Forward Fully Connected Network (FF-FCN)**.
  + Data was reshaped into a 28x28 format for **Convolutional Neural Networks (CNN).**

### Dimensionality Reduction:

* + **Principal Component Analysis (PCA)** was used to reduce the dimensionality of the dataset while preserving most of the variance, significantly improving training time.

### Data Augmentation:

* + New samples were generated through transformations like **rotation**, **flipping**, and **scaling** to improve model generalization.

## Data Visualization

A sample of features and labels was visualized to understand the dataset better. The visualization showed pixel values ranging from 0 (white) to 255 (black), with digits clearly distinguishable.

## Model Selection

Several machine learning models were evaluated for the classification task:

#### 1. Logistic Regression:

Mathematically less complex and faster for large datasets.

Achieved an overall accuracy of **90.96%.**

#### 2. Support Vector Classifier (SVC):

Specifically designed for classification tasks.

Achieved the highest accuracy of **98.65**%.

#### 3. HistGradientBoostingClassifier:

A fast gradient boosting framework based on decision trees.

Achieved an accuracy of **97.17%.**

#### 4. KNeighborsClassifier:

A simple and popular classifier.

Achieved an accuracy of **98.04%.**

#### 5. Feed-Forward Fully Connected Network (FF-FCN):

A neural network with dense layers.

Achieved an accuracy of **96.98%.**

6. Convolutional Neural Network (CNN):

Best suited for image recognition due to its ability to learn non-linear relationships.

Achieved an accuracy of **98%** with an average loss of **0.076**.

## Hyperparameter Tuning

Grid search was used to optimize hyperparameters for each model.

The best parameters for each model were:

* + **Logistic Regression**: ‘C=10’, ‘solver='saga'‘, ‘max\_iter=100’
  + **SVC**: ‘C=10’, ‘kernel='rbf'
  + **HistGradientBoostingClassifier**: ‘learning\_rate=0.1’, ‘max\_iter=1000’, ‘max\_leaf\_nodes=31’
  + **KNeighborsClassifier**: ‘n\_neighbors=5’, ‘weights='distance'‘, ‘metric='minkowski'‘

## Model Performance

* **SVC** performed the best with an accuracy of **98.65%** and an AUC of **0.9997**.
* **CNN** also performed exceptionally well with an accuracy of **98%** and an average loss of **0.076**.
* **FF-FCN** achieved an accuracy of **96.98%,** while **Logistic Regression** had the lowest accuracy among the models at **90.96%**.

## Challenges and Findings

1. **Training Time**:

* Dimensionality reduction using PCA significantly improved training time. For example, SVM training time reduced from ~20 minutes to ~3 minutes.
* **KNN** was the fastest to train, taking less than 1 second.

1. **Model Improvements**:

* **HistGradientBoostingClassifier** outperformed **GradientBoostingClassifier** for large datasets.
* Data augmentation techniques like rotation, flipping, and scaling improved model generalization but increased computational load.

1. **Debugging**:
   * + Hardcoding dimensions was avoided, and shapes were printed at each step to ensure compatibility between layers.
     + Issues like non-convergence in **Logistic Regression** (using ‘lbfgs’) and inconsistencies in **SVC** (when ‘probability=True’ was not enabled) were addressed.
2. **Misclassification Patterns**:

* Common misclassifications occurred between digits like **4 vs 9** and **3 vs 5**.

## Conclusion

* **SVC** and **CNN** were the best-performing models, achieving accuracies of **98.65%** and **98%**, respectively.
* **CNN** is particularly well-suited for image recognition tasks due to its ability to learn non-linear relationships.
* **Logistic Regression** with the ‘saga’ solver was faster than the default ‘lbfgs’ solver by approximately 3 times.
* Dimensionality reduction techniques like PCA and data augmentation methods improved model performance and generalization.

## Future Work

* Explore more advanced data augmentation techniques.
* Experiment with deeper neural network architectures.
* Investigate ensemble methods to combine the strengths of multiple models.

## Acknowledgments

Special thanks to the **scikit-learn**, **PIL**, and **PyTorch** libraries for providing the tools necessary to complete this project.

## End of Report

This project successfully demonstrated the application of various machine learning models to the task of handwritten digit classification, with **SVC** and **CNN** emerging as the top performers.