Flowchart Final and observation report

Flowchart:

1. Import required libraries
2. Load train.csv and test.csv
3. Data exploration
4. Dataset size + class distribution
5. Display sample images 28x28
6. Check missing values
7. Normalize pixels 0-255 to 0-1
8. Define tunable hyperparameters
9. If PCA Enabled convert the 784 dimensions to n_components dimensions
10. If PCA is not enabled Use raw 784 pixel features
11. Training of all models(KNN, SVM and Decision Tree)
12. Evaluations of all the models on the test dataset
13. Calculating the accuracy of each model
14. Confusion matrix heatmaps and classification report
15. Misclassified analysis
16. Save misclassified images
17. Save confusion-pair summary CSV
18. Final report

Observations and Conclusion:

This project implements a classical machine learning pipeline to classify handwritten digits (0–9) using MNIST CSV images. The dataset was explored by viewing sample images and class distribution, and pixel values were normalized to the range 0–1. PCA was optionally applied to reduce dimensionality, improving computational efficiency for KNN and SVM. KNN was implemented fully from scratch using Euclidean distance and majority voting. A vectorized “Fast KNN” version was used instead of a loop-based simple implementation to reduce runtime while keeping the same algorithm. Simple KNN was slow on the training (took around 50 minutes for training) dataset so a FastKNN version was implemented for faster training(took few minutes only) as the training dataset was large(60000).

Among the models, SVM generally achieved the best accuracy due to stronger decision boundaries, while Decision Tree trained quickly but produced more errors due to overfitting.

The use of subset hyperparameter was not implemented(it was kept Fasle) during the final analysis for both SVM or KNN as it reduced the training samples and reduced the accuracy of the models and caused them to misclassify more digits.

During the training of models, both Simple and Fast KNN architecure were trained with PCA. Even though PCA was enabled the simple version of KNN took longer time to train(approx. 50 mins) while the fast KNN trained within minutes. there was no noticeable difference between the accuracies of these two models with PCA enabled. The only difference was time so in this notebook only the fast KNN model is used.

Decision tree model performed the least accurately amongst these 3 models. Hyperparameter tuning was done to see the effect of various configurations. The best configuration found was the final one on which the number of misclassified digits were recorded.

Misclassifications frequently occurred between visually similar digits such as 4 and 9 or 3 and 5 or 3 and 8 and so on. Misclassification of digits usually happened between digits which were poorly written and the ones which have similar strokes while writing. The voting ensemble improved stability by combining predictions from multiple models. Further improvements could include tuning PCA components, optimizing hyperparameters, and applying weighted voting.

This notebook was run using kaggle and the dataset was directly attached to the input of this notebook. The dataset was alredy available as training and test data set in csv format.

Link: https://www.kaggle.com/datasets/oddrationale/mnist-in-csv