Efecan Kasapoğlu

31038

HW1

Colab Notebook Link: https://colab.research.google.com/drive/1MfIWQqreCpAqWM2b9rWeet4AErRd2dlP?usp=sharing

1. Introduction

In this project I worked with the MNIST dataset, that contains handwritten digits from 0 to 9. This dataset contains 60,000 training images and 10,000 test images. To analyze this dataset, I implemented k-Nearest Neighbors and Decision Tree algorithms.

2. Dataset and Preprocessing

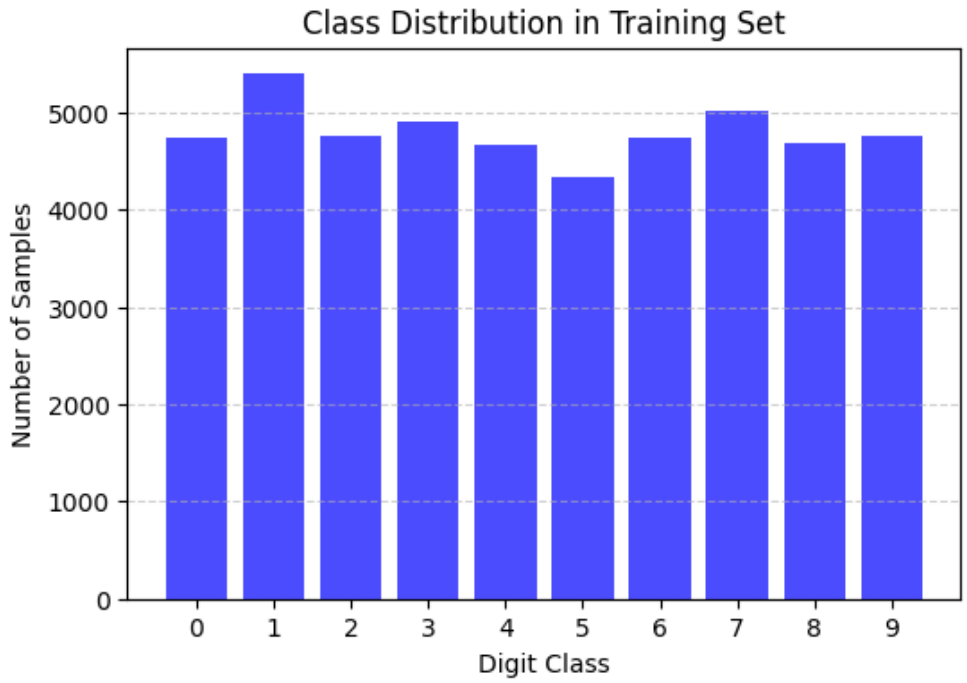
2.1 Data Loading

This dataset contains 28x28 grayscale images of handwritten digits from 0 to 9. I used 80% (48,000) of the dataset for the training set and the remaining 20% (12,000) for the validation set. Moreover, there are no modifications on the given test set. The shape of the dataset as follows:

- Training Data Shape: (48000,28,28)
- Validation Data Shape: (12000,28,28)
- Test Data Shape: (10000,28,28)

2.2 Data Analysis

I calculated the total number of samples for each digit. Moreover, I drew a bar chart to see the distribution. As a result, there does not seem to be a big imbalance between the classes.



Class Distribution: {0: 4738, 1: 5394, 2: 4766, 3: 4905, 4: 4674, 5: 4337, 6: 4734, 7: 5012, 8: 4681, 9: 4759}

The mean and the standard deviation of this dataset is calculated to analyze the pixel value distribution. The mean is **33.35111325467687** and the standard deviation is **78.60129135177951**. The mean shows that most pixels are darker. Standard deviation indicates that there is a wide variation in pixel intensities.

I also created subplots to visualize the dataset and showed one sample image for each digit.

2.3 Data Preprocessing

To prepare the dataset for training, normalization was applied to scale pixel values into the range [0,1]. At the beginning the value of the pixel was between 0 to 255; however, after the normalization the values are between 0 to 1. Each pixel value is divided by 255. After the normalization basic statistics recalculated. Mean pixel value is **0.13078867943010575** and standard deviation is **0.30824035824227297**.

3. k-NN Classifier

3.1 Model Initialization and Hyperparameter Tuning

For the experiment, k-NN classifier was initialized. To determine the optimal K value, the model was trained and evaluated using different values of K (1,3,5,7,9). Moreover, Euclidean distance and Manhattan distance metrics were tested. The combination of K and distance metric evaluated on the validation test.

k=1, metric=euclidean -> Validation Accuracy: 0.9704

k=1, metric=manhattan -> Validation Accuracy: 0.9644

k=3, metric=euclidean -> Validation Accuracy: 0.9703

k=3, metric=manhattan -> Validation Accuracy: 0.9633

k=5, metric=euclidean -> Validation Accuracy: 0.9675

k=5, metric=manhattan -> Validation Accuracy: 0.9620

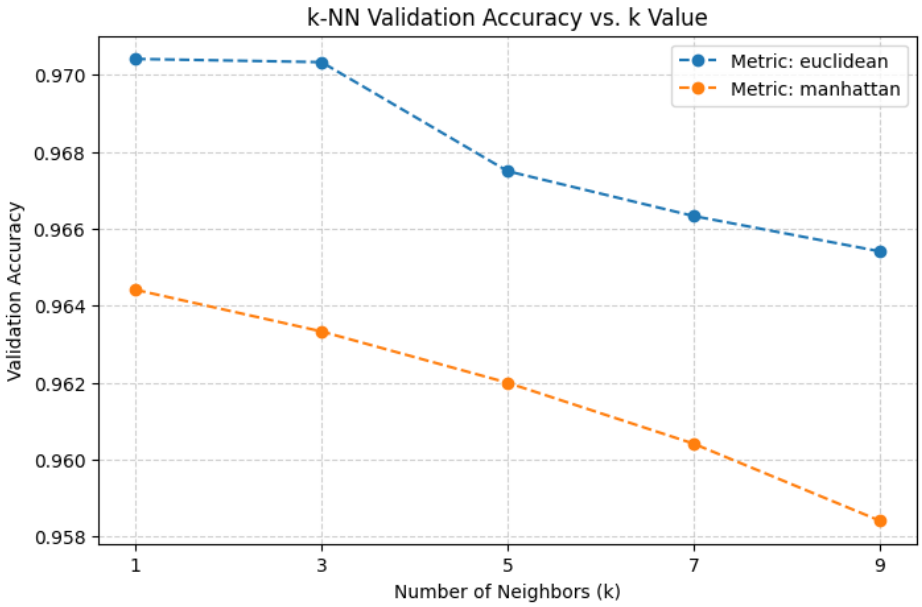
k=7, metric=euclidean -> Validation Accuracy: 0.9663

k=7, metric=manhattan -> Validation Accuracy: 0.9604

k=9, metric=euclidean -> Validation Accuracy: 0.9654

k=9, metric=manhattan -> Validation Accuracy: 0.9584

As a result, the highest validation accuracy was achieved with k=1, Euclidean distance and 97,04% validation accuracy.



3.2 Final Model Training and Evaluation

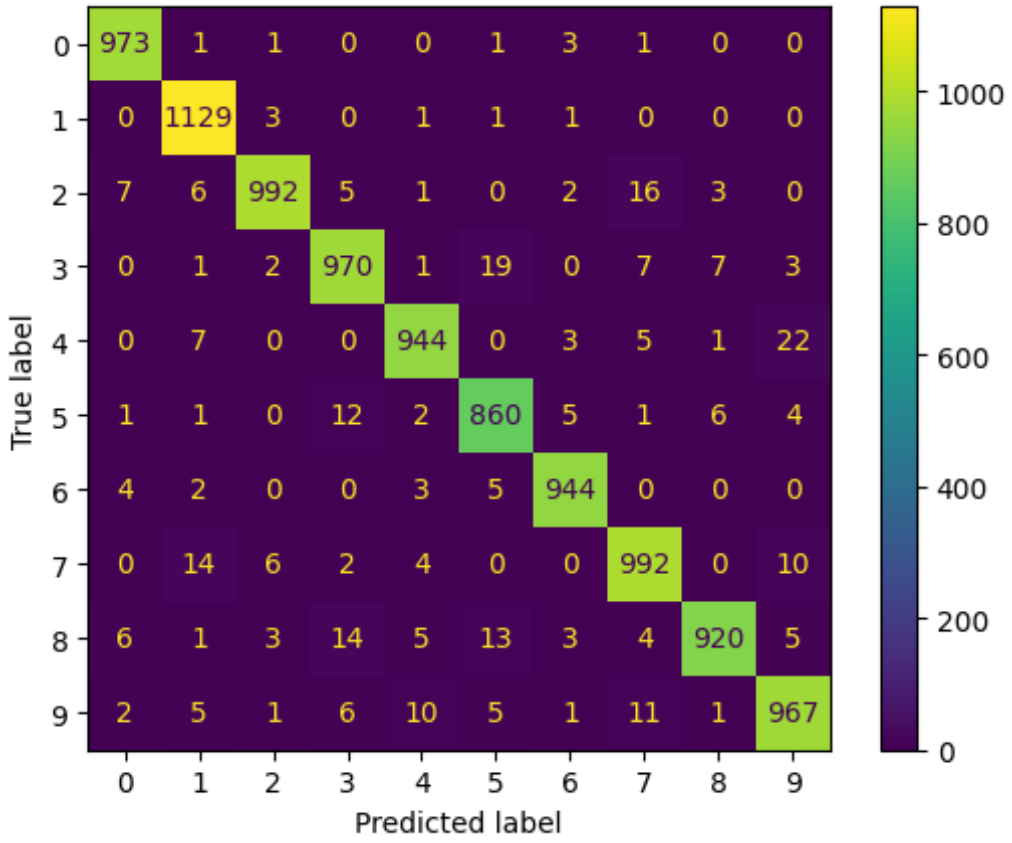
After finding the optimal K value (K=1) and the distance metric (Euclidean), k-NN classifier was retrained using the optimal K value and the distance metric. The final model was evaluated on the test set using Accuracy, Precision, Recall and F1 Score.

Classification Report for k-NN:

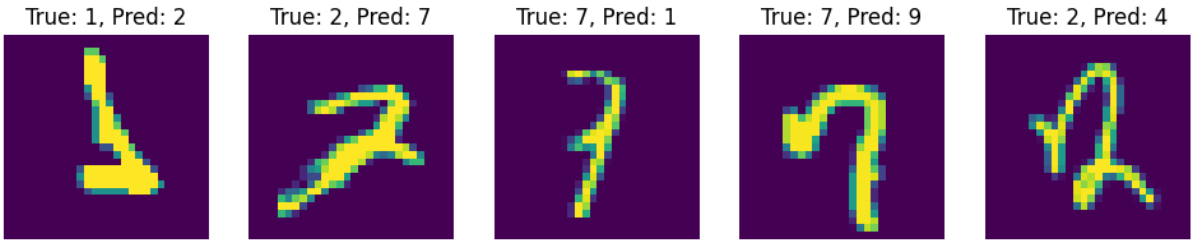
	precision	recall	f1-score	support
0	0.98	0.99	0.99	980
1	0.97	0.99	0.98	1135
2	0.98	0.96	0.97	1032
3	0.96	0.96	0.96	1010
4	0.97	0.96	0.97	982
5	0.95	0.96	0.96	892
6	0.98	0.99	0.98	958
7	0.96	0.96	0.96	1028
8	0.98	0.94	0.96	974
9	0.96	0.96	0.96	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

Following this training, the model was evaluated on the test set, achieving a test accuracy of 96.91% with the best hyperparameters. The model mostly achieved strong results; however, some digits were misclassified.

Confusion matrix was generated to analyze errors. The numbers shows the correct and incorrect predictions for each digit.



By analyzing confusion matrix, most misclassified digits are 3, 9 and 8. 3 is mostly confused with 5. 9 is mostly confused with 4 and 7. 8 is mostly confused with 2 and 5**.** Five random misclassified examples were selected and displayed.



4. Decision Tree Classifier

4.1 Model Training and Hyperparameter Tuning

The Decision Tree classifier was trained to further analysis. Model's performance depends on the hyperparameter selection. To find the best performing model, grid search with the cross validation 3 is applied. The max depth values of 2, 5, and 10 were tested, along with min samples split values of 2 and 5.

```
All Hyperparameter Combinations and Their Accuracies:
Max Depth:  2, Min Samples Split:  2 → Validation Accuracy: 0.3399
Max Depth:  2, Min Samples Split:  5 → Validation Accuracy: 0.3399
Max Depth:  5, Min Samples Split:  2 → Validation Accuracy: 0.6709
Max Depth:  5, Min Samples Split:  5 → Validation Accuracy: 0.6709
Max Depth: 10, Min Samples Split:  2 → Validation Accuracy: 0.8492
Max Depth: 10, Min Samples Split:  5 → Validation Accuracy: 0.8496

Best Hyperparameters: Max Depth: 10, Min Samples Split: 5
Best Cross-Validation Accuracy: 0.8496
```

The highest validation accuracy was achieved with max depth=10 and min samples split=5, resulting in a best cross-validation accuracy of 0.8496.

4.2 Evaluation

The performance of the Decision Tree classifier was measured using accuracy, precision, recall, and F1-score. The final test accuracy achieved was 86.57%.

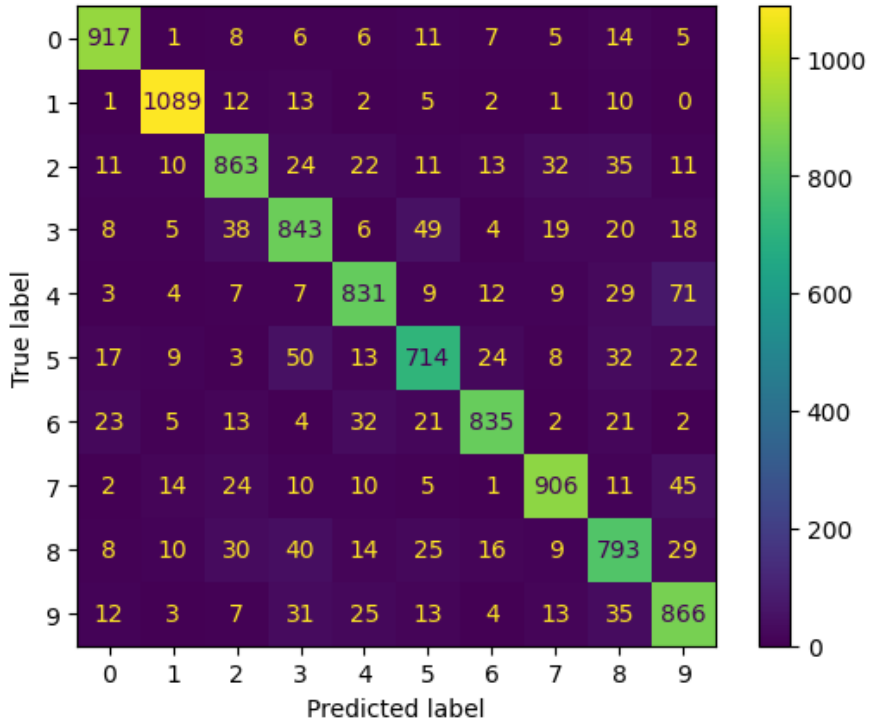
```
Test Accuracy with Best Hyperparameters: 0.8657

Classification Report for Decision Tree:
              precision    recall  f1-score   support

           0       0.92      0.94      0.93       980
           1       0.95      0.96      0.95      1135
           2       0.86      0.84      0.85      1032
           3       0.82      0.83      0.83      1010
           4       0.86      0.85      0.86       982
           5       0.83      0.80      0.81       892
           6       0.91      0.87      0.89       958
           7       0.90      0.88      0.89      1028
           8       0.79      0.81      0.80       974
           9       0.81      0.86      0.83      1009

    accuracy                           0.87     10000
   macro avg       0.86      0.86      0.86     10000
weighted avg       0.87      0.87      0.87     10000
```

For further analysis, confusion matrix provides a detailed analysis of errors. Some digits are more frequently misclassified than others.



By analyzing confusion matrix, most misclassified digits are 2, 5, and 8. 2 is mostly confused with 7 and 8. 5 is mostly confused with 3. 8 is mostly confused with 3.

The ROC curve was plotted for each digit on a single plot. The ROC curve provides a clear visualization of which digits are well-classified and which ones cause difficulties. Higher AUC scores indicate that most digits are well-separated by the classifier. However, digits 8,2, and 5 have lower AUC scores, suggesting they are more challenging to classify.