**HANDWRITTEN TEXT RECOGNITION**

**USING KNN**

ELC Activity

**Submitted by**

Name: GARIMA

Roll Number: 102217021

Subgroup: CS1

**AIM:**

Train your model using K-Nearest Neighbor Algorithm with having values of K as {2,4,5,6,7,10}, over data.csv file provided. The Train and Test split of the data should be in the ratio of 60:40, 70:30, 75:25, 80:20, 90:10, 95:5. Evaluate the performance of the model over test data for all these scenarios (36 cases), and submit the single jupyter notebook, having one of the scenario implemented (and rest in comments), and a single pdf file containing the results of all these scenarios (Accuracy, and Confusion Matrix), also your analysis regarding the dependency of the performance of model over training-testing split and k value.

**SOLUTION:**

For solving the Handwritten Text Recognition problem using K-Nearest Neighbors (KNN), I utilized the scikit-learn library to implement the model and evaluate its performance. Scikit-learn is one of the most useful libraries for machine learning in Python, offering efficient tools for classification, regression, clustering, and dimensionality reduction. In this implementation, I evaluated the model using five different values for KK (2, 4, 5, 6, 7, and 10) across various training-testing splits (0.4, 0.3, 0.25, 0.2, 0.1, and 0.05). The model's performance was assessed using accuracy, precision, recall, F1 score, and the confusion matrix. This comprehensive evaluation helped in understanding the dependency of the model's performance on the chosen KK values and the training-testing split, ensuring a robust and reliable handwritten text recognition system.

**ANALYSIS:**

Based on the provided data, here is an analysis of how the model's performance, as measured by accuracy, depends on the training-testing split and the value of KK in a K-Nearest Neighbors (KNN) classifier:

### **General Observations:**

1. Test Size Impact:
   * As the test size decreases (i.e., more data is used for training), the accuracy tends to increase slightly. This trend is visible across different values of KK.
   * For example, the highest accuracy values are often observed with a test size of 0.1 and 0.05.
2. K Value Impact:
   * The accuracy varies with different values of KK, typically showing a peak at certain values and then slightly decreasing or plateauing.
   * Generally, lower values of KK (like 2 or 4) seem to perform slightly worse than moderate values (like 5), but performance doesn't drastically change for higher values (like 10).

### **Detailed Observations by Test Size:**

1. Test Size: 0.4:
   * Accuracy improves from 0.9543 (K=2) to a peak at 0.9616 (K=5), then slightly drops.
   * Moderate K values (4-6) yield the highest accuracies around 0.961.
2. Test Size: 0.3:
   * Accuracy improves from 0.9565 (K=2) to a peak at 0.9640 (K=4), then slightly drops.
   * Again, moderate K values (4-6) are the best performers, with K=4 achieving the highest accuracy of 0.964.
3. Test Size: 0.25:
   * Accuracy improves from 0.9584 (K=2) to a peak at 0.9641 (K=4), then slightly drops.
   * Moderate K values (4-5) yield the highest accuracies, with K=4 achieving 0.9641.
4. Test Size: 0.2:
   * Accuracy improves from 0.9577 (K=2) to a peak at 0.9637 (K=5), then slightly drops.
   * Moderate K values (4-5) are the best performers.
5. Test Size: 0.1:
   * Accuracy improves from 0.9612 (K=2) to a peak at 0.9657 (K=4 and 5), then slightly drops.
   * Moderate K values (4-7) yield the highest accuracies around 0.965.
6. Test Size: 0.05:
   * Accuracy starts at 0.9581 (K=2), peaks at 0.9648 (K=5), then slightly drops.
   * K=5 achieves the highest accuracy of 0.9648.

### **CONCLUSION:**

1. Optimal Test Size:
   * Smaller test sizes generally lead to slightly higher accuracies, indicating that more training data improves the model's performance.
2. Optimal K Value:
   * Moderate K values (around 4-6) tend to perform best across different test sizes, with slight variations.
   * K=4 and K=5 frequently show the highest accuracies across various test sizes.
3. Performance Trade-offs:
   * There's a balance to be struck between the amount of data used for training and the complexity parameter KK. While increasing training data (smaller test sizes) improves accuracy, an optimal KK around 4-6 ensures the best performance.
   * The performance doesn't drop drastically with higher KK values (like 10), but the slight decrease suggests over-smoothing, which can be less effective than moderate KK values.

In summary, for the given data, a moderate KK value (4-6) combined with a smaller test size (0.1 to 0.25) provides the best model performance.