Name: Akshita Jain
Roll No:102103837
Group: 2CO28

Project Report
ML model on Handwriting Recognition using KNN

The K-nearest neighbors (KNN) algorithm is a type of supervised learning algorithm that can be used for classification or regression tasks. In this case, we are using it for a classification task to recognize handwritten digits. The algorithm works by finding the K nearest neighbors to a given point in the feature space, where K is a hyperparameter that we can set. The class of the new point is then determined by the majority class among its K nearest neighbors.

STEPS:

1. Import the necessary libraries:
   I needed Pandas, Numpy, Matplotlib and scikit-learn

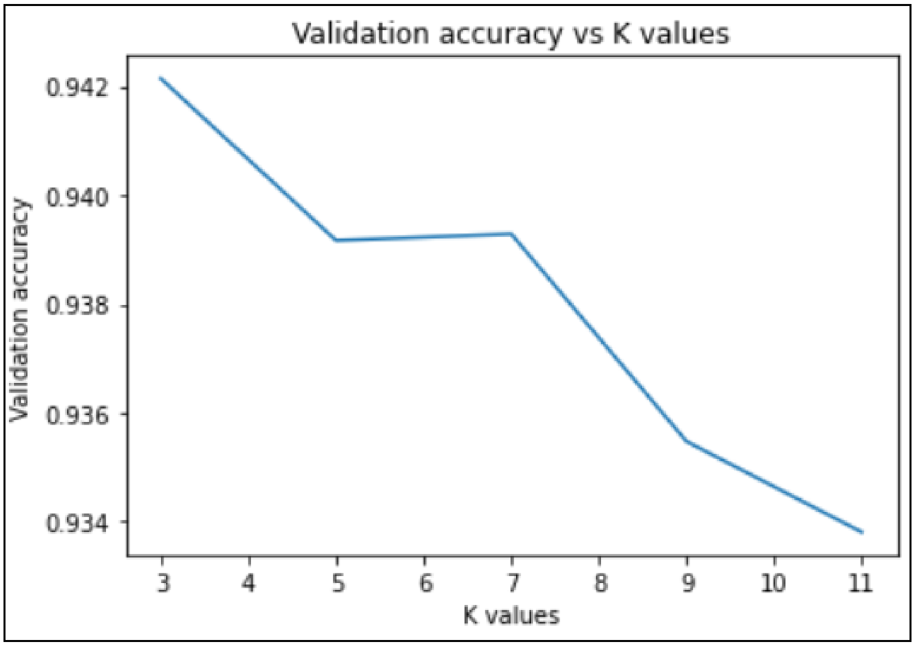
2. Load the training dataset: train.csv using pandas:
   I first uploaded the datasets on my google drive and then mounted my drive from where I loaded my dataset.

3. Split the dataset into training and validation sets.

4. Preprocess the data by scaling the features.

5. Train KNN classifier with K = 3, 5, 7, 9, 11 using the training set.

6. Evaluate the model on the validation set and record the accuracy.



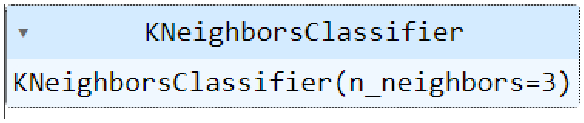
From the graph, it is clear that as we move from 3 to 11, our validation accuracy decreases.

7. Select the best K value based on the validation set performance.

```
Best K value: 3
```

In my case, the best K value comes out to be 3

8. Train a final KNN classifier on the entire training set using the best K value.

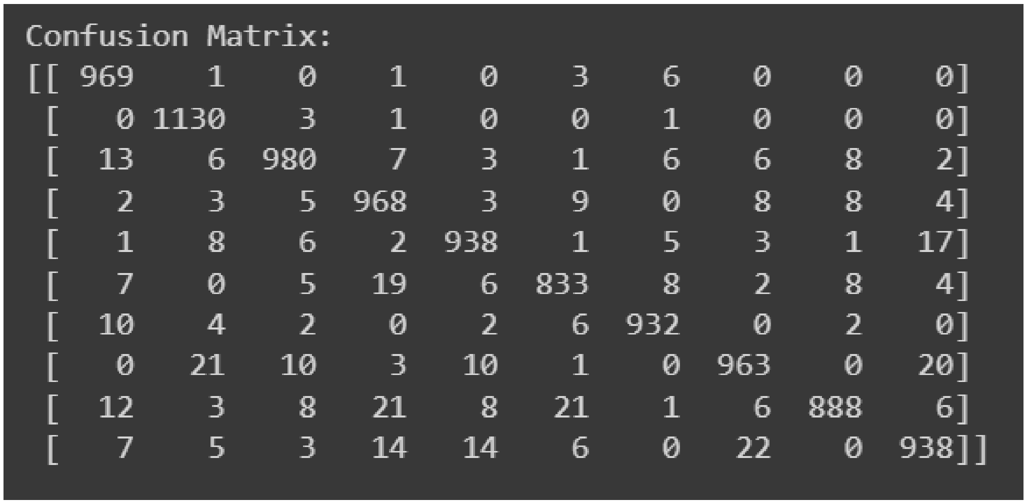


```
▾ KNeighborsClassifier
KNeighborsClassifier(n_neighbors=3)
```

9. Load the test dataset: test.csv using pandas.

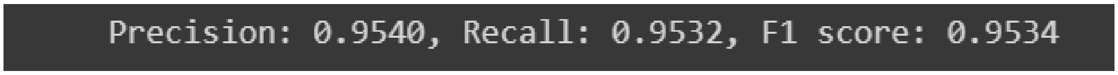
10. Preprocess the test data by scaling the features.

11. Test the trained model on the test dataset and generate a confusion matrix:
    A confusion matrix is a table that is used to evaluate the performance of a classification model on a set of test data for which the true values are known. The matrix compares the predicted class labels to the true class labels and displays the number of true positives, true negatives, false positives, and false negatives.



```
Confusion Matrix:
[[ 969    1    0    1    0    3    6    0    0    0]
 [   0 1130    3    1    0    0    1    0    0    0]
 [  13    6  980    7    3    1    6    6    8    2]
 [   2    3    5  968    3    9    0    8    8    4]
 [   1    8    6    2  938    1    5    3    1   17]
 [   7    0    5   19    6  833    8    2    8    4]
 [  10    4    2    0    2    6  932    0    2    0]
 [   0   21   10    3   10    1    0  963    0   20]
 [  12    3    8   21    8   21    1    6  888    6]
 [   7    5    3   14   14    6    0   22    0  938]]
```

12. Calculate the precision, recall, and F1 score based on the confusion matrix.



```
Precision: 0.9540, Recall: 0.9532, F1 score: 0.9534
```

Precision, recall, and F1 score are metrics used to evaluate the performance of a classification model.

Precision is the ratio of true positive predictions to the total number of positive predictions made by the model. It measures how accurate the model's positive predictions are.

Recall is the ratio of true positive predictions to the total number of actual positive instances in the test data. It measures how well the model identifies all positive instances.

F1 score is the harmonic mean of precision and recall. It provides a single score that balances both precision and recall.

In this case, Precision: 0.9540, Recall: 0.9532, and F1 score: 0.9534 suggest that the model has good performance in both identifying true positives and avoiding false positives. These values are relatively close to each other, indicating that the model has a balanced performance in terms of precision and recall.