Digit Recognition

Sourabh Ghurye, Nikhil Vyas, Utkarsh Mall
130050001, 130050023, 130050037

Problem Statement

Given a Training Data with images and corresponding digits to learn to classify new data among the characters.

This is a subcase of OCR which is meant to recognise characters. OCR is useful as a form of data entry from printed paper records as well as to digitalize printed texts.

Overview

We use MNIST dataset for training and testing. The input is a 28*28 (784) grayscale image i.e. each pixel has an associated float in between 0 and 1.

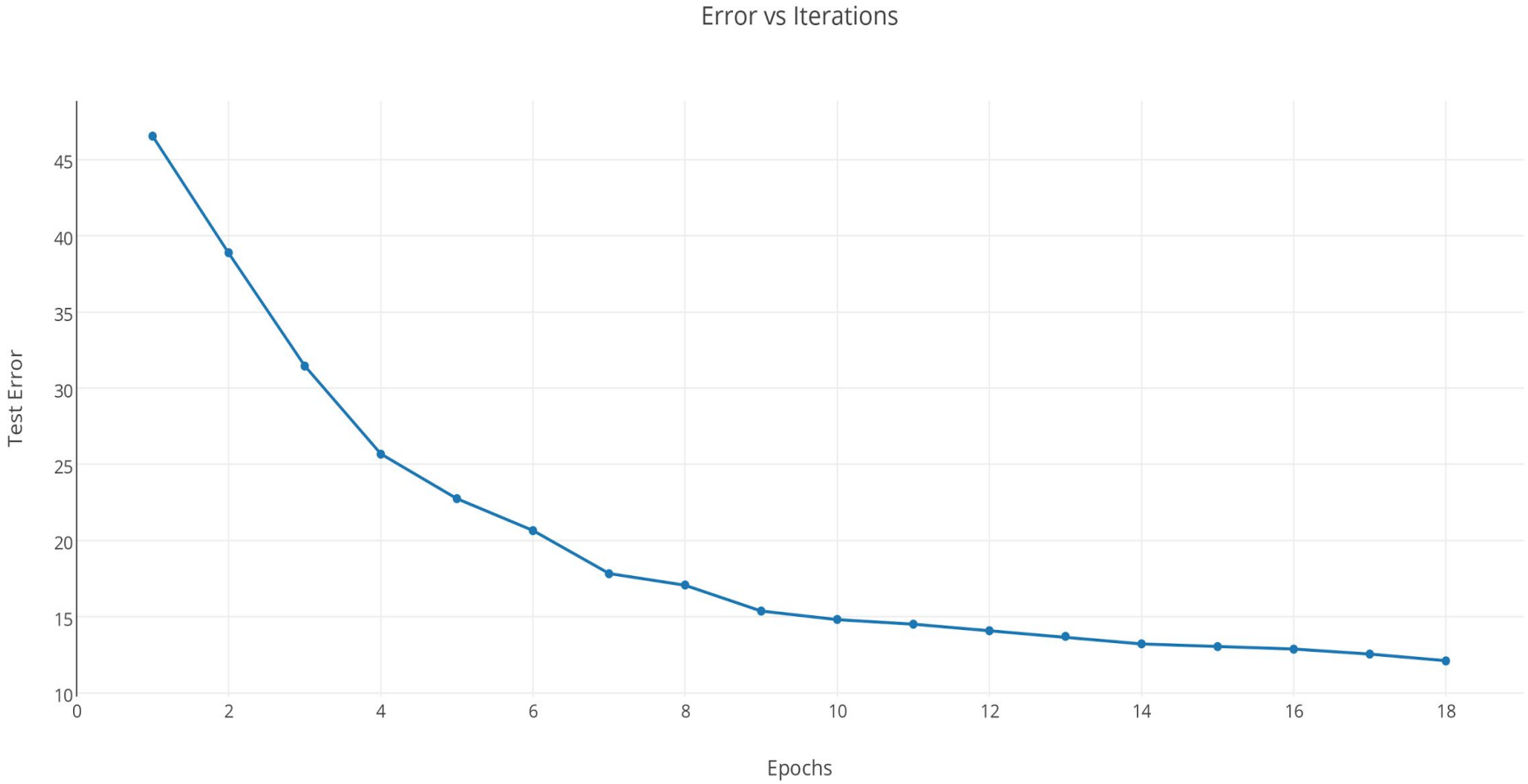
The approaches implement for solving the problem are:

a. Neural Networks
b. Support Vector Machines
c. k-Nearest Neighbor Classifier

Neural Network

Neural Network is implemented through PyBrain library. The structure of the neural network is (784, 30, 10) i.e. 784 (28*28) input nodes, 30 hidden layer neurons and 10 output layer neurons. Each output layer neuron represents a digit.

Data on error on test data vs number of iterations was measured till it converged.


Error vs Iterations
Test Error
45
40
35
30
25
20
15
10
0
2
4
6
8
10
12
14
16
18
Epochs

Support Vector Machines

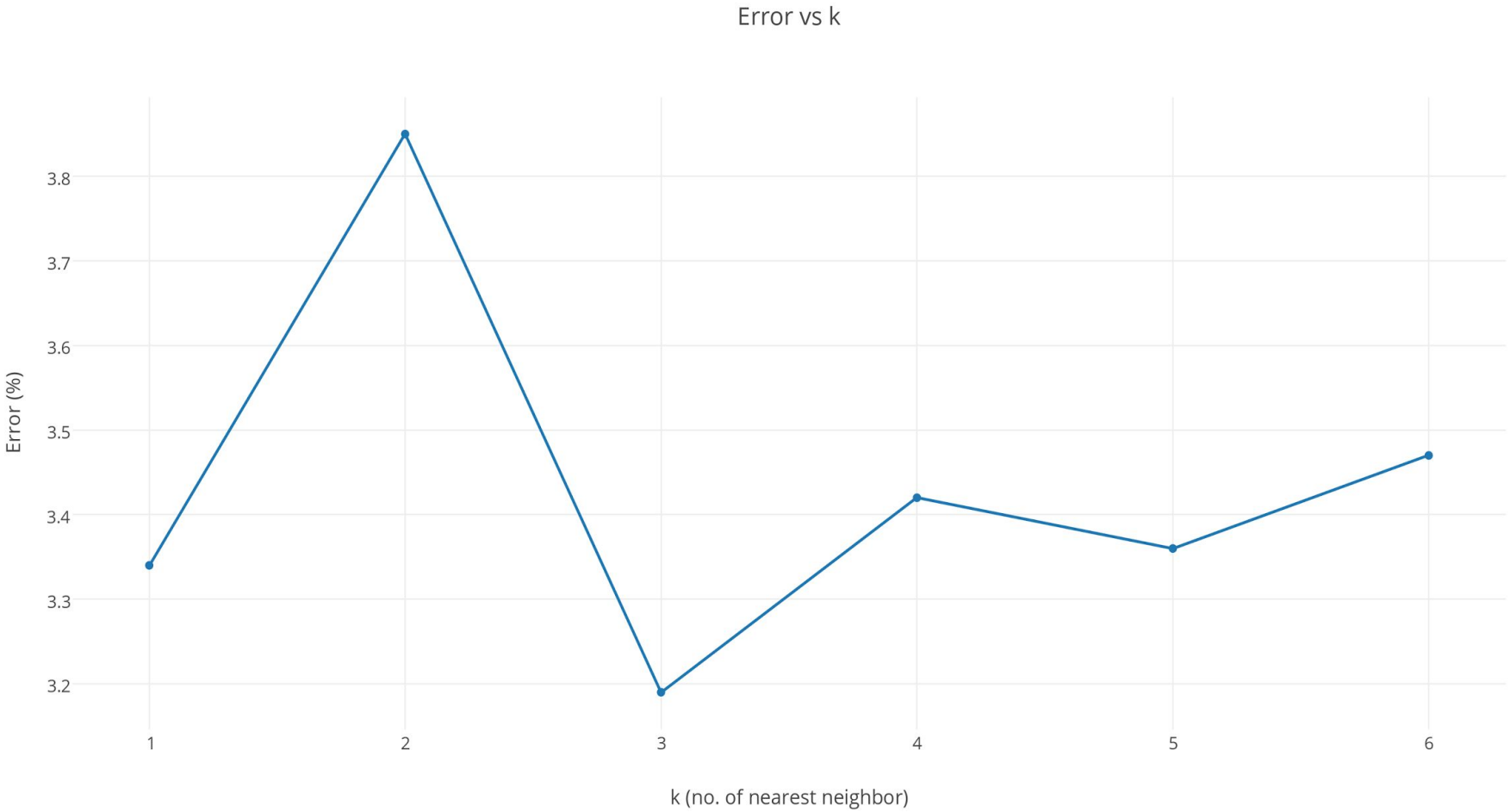
Support Vector Machines (SVM) was implemented using scikit-learn library for python.

The error measured on the test data was about 3.3%, lower than the final error of the neural network.

k-Nearest Neighbor Classifier

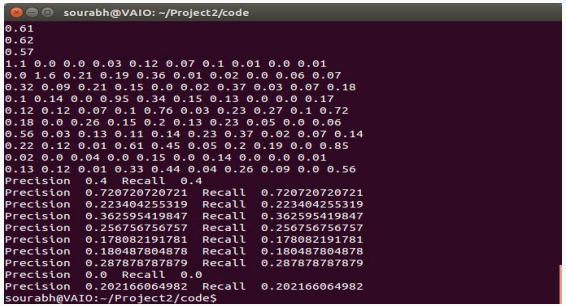
Nearest Neighbor classifier was implemented using scikit-learn library for python.

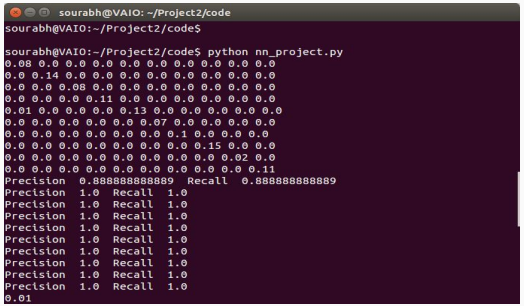
The error on the training was recorded for different numbers on nearest neighbors. It is in the form of peaks (on even numbers) and troughs (on odd numbers) with the lowest error being 3.19 % at k = 3. This is as even number add a random error compared to the odd numbers.

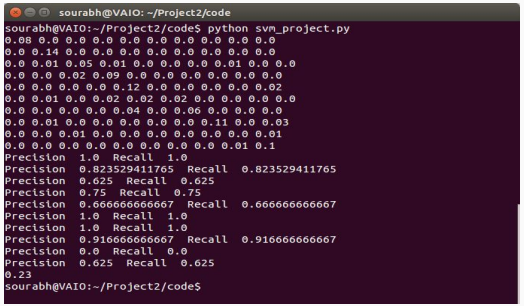

Error vs k
Error (%)
3.8
3.7
3.6
3.5
3.4
3.3
3.2
1
2
3
4
5
6
k (no. of nearest neighbor)

Precision and Recall

We calculated precision, recall and confusion matrix for all three classifiers.







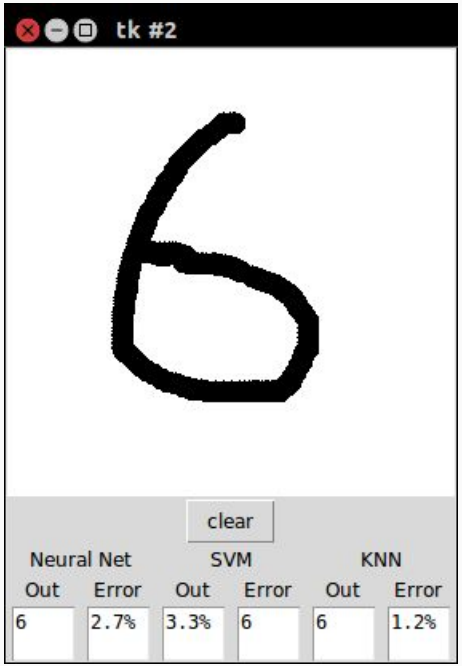
Neural Net

svm

KNN

User Interface

We created user interface that enables user to draw digits and get recognized output using the three different methods.



Conclusion and future work

Amongst the three implementations k-Nearest Neighbor performed the best followed by Support Vector Machines with Neural Neural at last.

This was mainly as the neural network was unoptimized. As a continuation of this work a convolutional neural network with sparsity (SCNNs) can be implemented. These are designed so as to exploit the structure of an image and give much better results than both Support Vector Machines and k-Nearest Neighbor.