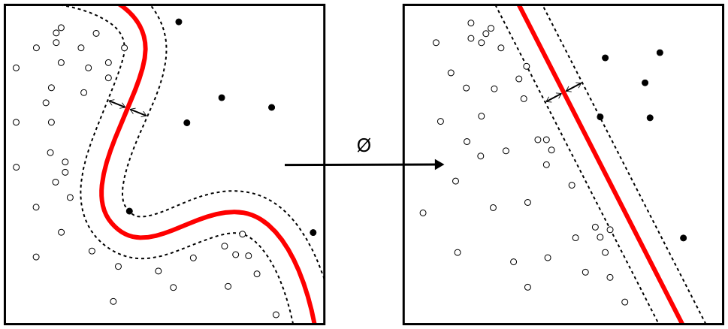
Algorithms

Out of the numerous algorithms present for handwritten digit recognition, we have chosen to research and analyze the best ones currently present.

Random Forest Algorithm

Random forest is a method for classification task that creates multitude of decision trees while training and the class which is most occurring of all the classes is the output. Trees which have large depth have a tendency to produce output which overfit. Random forest helps in averaging multiple decision trees which are trained on different parts of same dataset. This helps in reducing the high variance in output which is generally produced in decision trees. Some features of random forest are as follows:

* Excellent Accuracy
* Efficient on large datasets
* Estimates important variable in a classification
* Handle thousands of input variable without variable deletion

**Random forest Pseudocode**

A training set S := (x1, y1), . . . ,(xn, yn), features F, and number of trees in forest B.

1 function RandomForest(S , F)

2 H ← ∅

3 for i ∈ 1, . . . , B do

4 S (i) ← A bootstrap sample from S

5 hi ← RandomizedTreeLearn(S (i) , F)

6 H ← H ∪ {hi}

7 end for

8 return H

9 end function

10 function RandomizedTreeLearn(S , F)

11 At each node:

12 f ← very small subset of F

13 Split on best feature in f

14 return The learned tree

15 end function

We select an ith sample S(i) using which we learn a decision tree. The algorithm is modified such that, at each node of the tree we randomly select some subset features. The node is then split on best feature f where f<F. Since the number of features are reduced, it drastically increases the speed of computation.

***k*-Nearest Neighbors Algorithm**

*k-*Nearest Neighbors are used in pattern recognition for classification and regression methods. Here k is the closest training example in the training datatset. The output is the majority class in the k nearest neighbors. If k=1, the output is the class of single nearest neighbor.

Steps in *k*-NN Algorithm

1. Determine the parameter K
2. Compute the Euclidean distance between the query instance and Training samples
3. Sort the distance computed
4. Determine the nearest neighbor using the sorted distance
5. Identify the class of *k* nearest neighbors
6. Most occurring class is the predicted value

**Support Vector Machine**

Support Vector Machines are supervised learning methods which are used for classification and regression analysis. They construct a single or a set of hyperplanes which can be used for classification. A data point in SVM is considered as a p-dimensional vector which can be separated using (p-1) dimensional hyperplane. Hyperplane which has maximum margin between two classes is generally chosen. If a set of data points are not linearly separable then they are mapped into higher dimensions which makes separation easier. This is known a kernel trick.

Figure 1: Kernel Machine

**Linear SVM**

Consider n points training dataset (x1,y1)…….(xn,yn) where y can 1 or -1 and each x is a p dimensional real vector. Any Hyperplane can be written as

Here, is the normal vector to the Hyperplane.

In order to identify the maximum-margin hyperplane, two hyperplane with maximum distance is identified which can separate two classes of data. Equations describing these hyperplanes are

Distance between these hyperplane is , hence inorder to maximize the distance .

**Nonlinear SVM**

Data sets that are not linearly separable we try to separate them by using kernel trick that converts them into higher dimensions as shown in Figure 1. It is comparatively easy to separate nonlinear data in higher dimensions.

**Artificial Neural Network**