

# A Comparative Study and Performance Analysis of Classification Techniques: Support Vector Machine, Neural Networks and Decision Trees

Kumarshankar Raychaudhuri<sup>(✉)</sup>, Manoj Kumar, and Sanjana Bhanu

Department of Computer Science, Ambedkar Institute of Advanced Communication Technologies and Research (AIAC&R), GGSIPU, Delhi 110031, India

ksrcl9@gmail.com, sanjanabhanu@gmail.com,  
manojgaur@yahoo.com

**Abstract.** A support vector machine (SVM) is a classification technique in the field of data mining, used for the classification of both linear as well as non-linear data. It learns the decision surface from two different classes of input samples and then performs analysis of new input samples. A neural network is able to learn without the explicit description of the problem or the need of a programmer. Another type of classification technique is the decision tree. In this paper, we are doing a comparative study of the above mentioned classification techniques by analyzing their performance on data sets. We will be comparing the inputs and the observed outputs.

**Keywords:** Support Vector Machine (SVM) · Artificial Neural Network (ANN) · Decision Trees (DT) · Classification · Hyperplane

## 1 Introduction

Classification is a data analysis technique, used to categorize data into different classes or to predict future trends in the data. Different classification techniques have been proposed by researchers in the areas of machine learning, pattern recognition, and statistics. Classification is carried out in two phases. During first phase, a classifier is developed by training with a pre-determined set of data inputs, using a classification algorithm. This is referred to as the “training phase” or “learning phase”. This type of learning called supervised learning as well. In the second phase, the classifier model, is used for classification. In this phase, another set of data, called the test data set is used. The test data set is prepared by arbitrarily selecting inputs from the general data set, that are not included in the training data set.

Support Vector Machine is an efficient classification technique. As proposed by Vapnik [1]. SVM’s is one of the best “off-the shelf” supervised learning algorithm. A special property of SVM is that it can maximize the relative geometric margin, which decreases the future test errors during classification. The SVM maps the input vectors

to a high dimensional feature space by constructing a Decision Boundary, also called “Optimal hyperplane” [2]. Two more hyperplanes are constructed parallelly on either sides of the maximal hyperplane, which ultimately classifies the input data into two different classes. The objective is to find the optimal hyperplane and the solution of the hyperplane is a combination of few input points also known as “Support Vectors.”

A neural network is an interconnected network of parallel, distributed information processing elements. The idea to develop artificial neural networks (ANNs) comes from the desire to develop artificial structures capable of doing sophisticated, and fast computations similar to the neurons present in the human brain [3]. It has been observed that SVM classifies with quite a higher accuracy than that of ANN, regardless of the input data set and the algorithm used to train the neural network [3].

Decision tree is another technique for classification. It is basically a binary tree, where the test on an attribute is denoted by an internal node, the outcome of the test denoted by the branch, and the class label is held by leaf node. One of the reason for the popularity of DT classifier is that it can be build without any knowledge about the domain or parameter setting [6].

The paper is divided into VI sections. Section 2 represents details of SVM. Section 3 deals with the Neural Networks. Decision Tree is represented in Sect. 4 and the comparative study of SVM, ANNs, and Decision Tree is described in Sects. 5 and 6 presents Analysis and Conclusion.

## 2 Support Vector Machines

The classification process in a SVM is done by forming a single or a set of hyperplanes in a high-dimensional feature space. The hyperplanes are defined as the set of points whose dot product with a vector in that space results in a constant. When linearly separable training data is used, two hyper planes are selected in a manner such that none of the points lie in between them, thus leading to the separation of the data. Then, the bounded area is called as the “margin” [1]. Hence, a margin can be defined as the width of the decision boundary, having no training points within. It can vary with the position and orientation of the separating hyperplane. There can be more than one hyperplanes between any two classes of input points in the feature space and the set of all such decision boundaries is known as the “Feasible Region.” As the margin keeps on increasing, the feasible region reduces and there reaches an instance, when all the regions will coincide and the margin can no longer be incremented, else, the feasible region will become zero. At this point, all those points, which touch the decision boundary are called the “Support vectors” and they are instrumental in controlling and influencing the decision boundary.

Maximizing the margin is a good idea, because a larger margin ensures that the future test samples are less likely to cross the hyperplane, leading to less errors. By Vapnik [1], it is proved below that there is a bound existing on the expected loss of future test samples also known as Risk, which is presented by the following equation:

Here, in Eq. (1)  $R(\omega)$  is the risk in the decision boundary, ( $R_{\text{train}}(\omega)$  is the error associated with the training data and  $h$  is the V.C dimension)

$$R(\omega) \leq R_{\text{train}}(\omega) + [f(h)/N]^{1/2} \quad (1)$$

This equation makes a prediction that in future, whatever risk that may occur from the decision boundary is bounded by the following two factors, as stated below:

- i. The training error,  $R_{\text{train}}(\omega)$
- ii. Function  $f(h)$ , where  $h$  is the V.C dimension, and  $f(h)$  is given by:

$$F(h) = h + h \log(2N) - h \log(h) - c \quad (2)$$

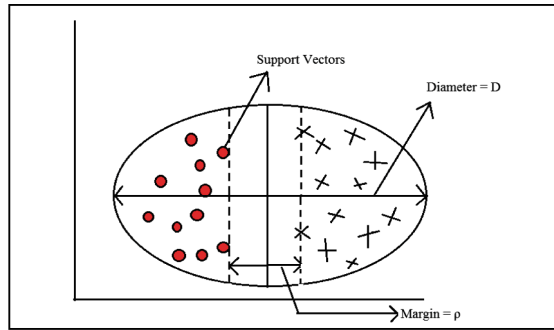
The above stated Eq. (2) gives a guarantee that the future test error of unknown samples (that have never been seen), will be less than the training error and a monotonically increasing function

$$[f(h)/N]^{1/2}$$

In order to reduce the test error, the margin needs to be maximized. For this, following two adjustments needs to be done:

- i. The training error ( $R_{\text{train}}$ ) should be kept low (say 0).
- ii. The V.C dimension,  $h$  should be minimized.

Figure 1 represents the smallest sphere (a circle in 2-dimensional space, as shown), with a diameter  $D$  that covers all the training points and  $\rho$  is taken to be the margin between the two sides of the hyperplane.



**Fig. 1.** Relative margin

$$\text{Relative Margin} = \rho/D$$

(The relative margin does not depend on the scale of points).

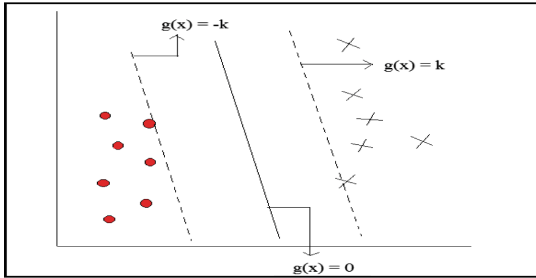
The main objective is to somehow maximize the relative margin. VC-Dimension (VC-D) is given by-

$$h \leq \left( \min \left( d, \frac{D^2}{\rho^2} \right) + 1 \right) \quad (3)$$

Here, in Eq. 3,  $d$  refers to the dimensionality of the set of points while  $D^2/\rho^2$  indicates the relative margin. Eq. 3 shows a relation between the VC-D and  $h$ . It shows that  $h$  is the minimum of  $d$  and the relative margin term. What people used to refer to as the “curse of Dimensionality”, the fact that when the value of  $d$  goes up, the test error also keeps going up and there is no way to control it, a scenario referred to as “over-fitting”, the VC-D provides a way to break this curse, where the VC-D would be bounded by the  $(D^2/\rho^2)$ , and hence it would be independent of the term  $d$  ( $d$  would not be needed anymore to find the value of  $h$ ). This forms one of the most important principles of “Maximum Margin Classification” [1].

Now, in order to maximize the relative margin:

Figure 2 shows the hyper plane with its boundaries, where  $g(x) = 0$  is the equation of the hyper plane.



**Fig. 2.** Hyper plane with decision boundaries

The decision boundary is represented by the equation:

$$W^T X + b = 0 \quad (4)$$

Here, in the above Eq. (4),  $W^T$  refers to the coefficient of the hyper plane,  $X$  refers to the training sample used,  $b$  refers to the bias constant, while  $d$  gives the value of the class notion.

In order to maximize  $k$ , following conditions needs to be satisfied:

$$W^T X_i + b \geq k, \quad \text{for } d_i = 1 \quad (5)$$

$$W^T X_i + b \leq -k, \quad \text{for } d_i = -1 \quad (6)$$

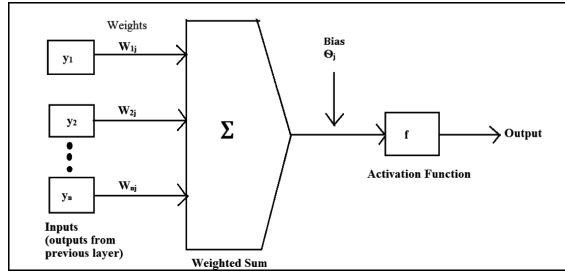
In the above Eqs. (5) and (6),  $d_i$  is the label of the samples, where it is +1 for one class of samples and -1 for another class.

### 3 Artificial Neural Networks

The primary aim for developing neural network is to simulate the working of a human brain, inside a computer system. ANN consists of neurons, which forms a network of interconnected cells.

A graphical representation of a neuron is given in the figure below:

Figure 3 describes the structure of a single neuron in the ANN, indicating the weights, which are outputs from the previous layer but act as input to this particular layer; a weighted sum is calculated and a non-linear activation function is applied to the net input [4].



**Fig. 3.** A single neuron structure

Given a unit  $j$  in an output layer, the net input  $I_j$  is given by:

$$I_j = \sum_i W_{ij} O_i + \Theta_j \quad (7)$$

In Eq. (10),  $W_{ij}$  is the weight of the connection from unit  $i$  in the previous layer to unit  $j$ ;  $O_i$  is the output of unit  $i$  from the previous layer, and  $\Theta_j$  is the bias of the unit.

The value of  $O_j$  is found using the logistic function as shown below:

$$O_j = [1 / (1 + e^{-I_j})] \quad (8)$$

In Eq. (11),  $I_j$  is the net input to the unit  $j$ . This function is also referred to as the “squashing function” because it maps a large input domain onto a smaller range of 0 to 1 [4]. Therefore,  $O_j(1 - O_j)$ , used in the previous equation is the derivative of logistic function.

Learning in ANNs can be broadly classified into three types: (i) Supervised Learning (ii) Unsupervised Learning and (iii) Reinforced Learning.

In Supervised Learning, the network computes a response to each of the input value, referred to as the “Output vector”. The output vector is then compared with the target vector. If the computed response differs from the target response significantly, then the weights of the network are adjusted according to the learning rule [5].

For a unit  $j$  in the output layer, the error  $E_j$  is calculated by:

$$E_j = O_j(1 - O_j)(T_j - O_j) \quad (9)$$

Here, in Eq. (12),  $O_j$  is the actual output of unit  $j$ , and  $T_j$  is the known target value of the training tuple.

The weights and the biases are updated to reflect the propagated errors. Weights are updated according to the following equation:

$$\Delta W_{ij} = (l)E_jO_i$$

Here,  $l$  is the learning rate, a constant whose value ranges between 0.0 and 1.0 [4]. Now, the adjusted weights are shown by the following equation:

Here, in Eq. (13),  $(s)$  indicates the current weights and  $(s+1)$  indicates the adjusted weights.

$$W_{ij}(s + 1) = W_{ij}(s) + \Delta W_{ij}(s) \quad (10)$$

After updating the weights, biases are updated according to the following equation:

$$\Delta \Theta_j = (l)E_j \quad (11)$$

$$\Theta_j = \Theta_j + \Delta \Theta_j \quad (12)$$

Here,  $\Delta \Theta_j$  is the change in the bias.

As shown above, the weights and biases are updated after the presentation of each tuple. This is referred to as “case updating” [4].

## 4 Decision Trees

Decision trees provide a simple yet powerful technique for classification of data. It is a flowchart-type structure where the test on an attribute is denoted by an internal node, the outcome of the test denoted by the branch, and the class label is held by leaf node. The error rate and accuracy rate in case of a decision tree are given by the following formulas:

$$\text{Error Rate} = \frac{\text{Total no. of misclassified points}}{\text{Total no. of data points}}$$

$$\text{Accuracy Rate} = (\text{Error Rate} - 1)$$

### Information Gain

Information gain is defined as the measure with which an attribute is chosen to perform the testing process at each node. ID3 uses information to select the attributes [4].

To calculate the information gain, it is essential first to calculate the entropy. Entropy, in general, is given by Eq. 13, as shown below:

$$\text{Entropy}(T) = \sum_{i=1}^n -\pi \log 2(\pi) \quad (13)$$

### Over Fitting Avoidance

In machine learning, “over fitting” is a general problem, especially in case of decision trees. Decision trees are trained to stop when all the data is well classified. In other words each branch is extended enough to classify data in that branch. To overcome the problem over-fitting, following two approaches are:

- i. Stop growing the tree before its perfection point.
- ii. Post-prune some of the branches of a fully grown tree.

## 5 Comparison of the Techniques Based on Training, Testing and Validation

The dataset description is as follows:

No. of features: 16  
 No. of instances: 435 samples  
 Training data size: 335 samples  
 Test data size: 50 samples  
 Validation data size: 50 samples

In this problem, we work with one of the UCI datasets on US congressional voting. The aim is to predict whether a US congressman (equivalent to a Member of Parliament in India) is Democrat or Republican based their voting pattern on various issues. The dataset provided has been split into 3 disjoint subsets: training data, validation data and data set. The simulation tool used for the purpose of carrying out the research work is MATLAB.

Comparison of efficiency measures between SVM and Neural Networks has been shown in Table 1 below, where C is a variable that controls the training error.

**Table 1.** Comparison of efficiency measures between SVM and neural networks

Cost	C = 1	C = 1000
SVM train	88.95%	94.92%
SVM test	82%	88%
SVM validate	82%	90%
ANN train	92.02%	62.31%
ANN test	82%	61%
ANN validate	89%	66%

Table 2 describes the efficiency of decision trees in training, testing, and validation when net error and information gain are used as best attributes for the growth of the decision trees. Information gain can be referred to as the measure with which an attribute is chosen to perform the testing process at each node. Training error and Testing errors are the two types of errors that can be made during the training and testing of the data sets. However, as the tree grows in size, the training error decreases. The testing error decreases in the beginning upto a certain point, with the increase in size of the tree, after which the training algorithm starts to cater to the noise in the data, which reduces its accuracy on the training data.

**Table 2.** Efficiency of decision tree with best attributes

Type	Fully grown (net error)	Fully grown (information gain)
Training	100%	100%
Test	90%	96%
Validation	86%	96%

Table 3 gives efficiency of decision trees in training, testing, and validation when the tree is pruned with net error and information gain to avoid over fitting [7]. To control the size of the tree, pruning is done, where the training data set is partitioned into the growing set and the pruning set. The growing set is used to build the tree, while the pruning set is used to find the approximate testing errors in the sub-trees, and the sub-tree having the minimum error is eventually selected as the decision tree.

**Table 3.** Efficiency of decision tree pruned to avoid over fitting

Type	Pruned (net error)	Pruned (information gain)
Training	88.95%	96.72%
Test	78%	98%
Validation	90%	96%

## 6 Analysis and Conclusion

In this paper, while comparing the classification techniques, it was observed that there has been variation in the results. It has been found that the accuracy differs in case of SVM and neural Networks (Table 1), when working on the same set of data. From Table 1, it is evident that when  $C = 1000$ , the results are in favor of Neural Networks, while the results change when  $C = 1$  i.e. in the latter case, SVM is able to classify the data with more accuracy as compared to Neural Networks.

In case of decision trees, a general trend is that a fully grown tree with net error criteria has a lower accuracy as compared to a fully grown tree with information gain criteria, as evident from Table 2. However, when tree is pruned using net error criteria, the accuracy is far less than the pruned tree with information gain criteria which is clearly shown in Table 3.



The above generated results depend on methods used to choose best attribute and criteria used to grow the tree.

Thus, it can be concluded that the classification techniques employed for the purpose of the research work, clearly depends on many factors. These factors play a major role for selecting one among many, as a result of which, it becomes difficult to state a technique better than the rest, in general. However, when a specific situation is given under certain circumstances, then it might be possible to prove one of the techniques to be superior than the others, which might fit the situation more perfectly than the rest.

## References

1. Vapnik, V.N.: An overview of statistical learning theory. *IEEE Trans. Neural Netw.* **10**, 988–999 (1999)
2. Durgesh, K.S., Lekha, B.: Data classification using support vector machine. *J. Theoret. Appl. Inf. Technol.* **12**, 1–7 (2010)
3. Byvatov, E., Fechner, U., Sadowski, J., Schneider, G.: Comparison of support vector machine and artificial neural network systems for drug/nondrug classification. *J. Chem. Inf. Comput. Sci.* **43**(6), 1882–1889 (2003)
4. Han, J., Kamber, M.: *Data Mining: Concepts and Techniques*. Morgan Kaufman Publishers, San Francisco (2007)
5. Nordbotten, S.: *Data Mining with Neural Networks*. Bergen, Norway (2006)
6. Kotsantis, S.B.: Decision Trees: a recent overview. *Artif. Intell. Rev.* **39**(4), 261–283 (2013). Springer
7. de Melo, G., Weikum, G.: Constructing and utilizing word nets using Statistical methods. *Lang. Resour. Eval.* **46**, 287–311 (2012). Springer

Advances in Computing and Data Sciences

First International Conference, ICACDS 2016,

Ghaziabad, India, November 11-12, 2016, Revised

Selected Papers

Singh, M.; Gupta, P.K.; Tyagi, V.; Sharma, A.; Oren, T.;

Grosky, W. (Eds.)

2017, XXIV, 638 p. 314 illus., Softcover

ISBN: 978-981-10-5426-6