Overview of Classification Technique

Paper Name – Term Paper

Submitted By

ARIJIT GHOSAL
Roll No. 001310505006

Under the supervision of

Dr. Sanjoy Kumar Saha, Associate Professor, CSE Dept.
and
Dr. Bibhas Chandra Dhara, Associate Professor, IT Dept.

JADAVPUR UNIVERSITY
KOLKATA - 700032

INTRODUCTION

The term pattern recognition refers to the task of placing some object to a correct class based on the measurements about the object. Usually this task is to be performed automatically with the help of computer. Objects to be recognized, measurements about the objects, and possible classes can be almost anything in the world. For this reason, there are very different pattern recognition tasks. A system that makes measurements about certain objects and thereafter classifies these objects is called a pattern recognition system. For example, a bottle recycling machine is a pattern recognition system. The customer inputs his/her bottles (and cans) into the machine, the machine recognizes the bottles, delivers them in proper containers, computes the amount of compensation for the customer and prints a receipt for the customer. A spam (junk-mail) filter is another example of pattern recognition systems. A spam filter recognizes automatically junk e-mails and places them in a different folder (e.g. /dev/null) than the user's inbox. The list of pattern recognition systems is almost endless. Pattern recognition has a number of applications ranging from medicine to speech recognition.

Pattern Recognition is a scientific technique to distinguish data or objects (in general pattern) into different categories or classes based on key features. Classes are groups having feature values similar. Pattern recognition algorithms generally aim to provide a reasonable answer for all possible inputs and to perform "most likely" matching of the inputs, taking into account their statistical variation. Some authors define a pattern as "*the opposite of a chaos; it is an entity, vaguely defined, that could be given a name*" or it can be said that a pattern can be a fingerprint image, a human face or a speech signal. The field of pattern recognition is concerned with the automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories.

Some pattern recognition tasks are everyday tasks (e.g. speech recognition) and some pattern recognition tasks are not-so-everyday tasks. However, although some of these tasks seem trivial for humans, it does not necessarily imply that the related pattern recognition problems would be easy. For example, it is very difficult to 'teach' a computer to read hand-written text. A part of the challenge follow because a letter 'A' written by a person B can look highly different than a letter 'A' written by another person. For this reason, it is worthwhile to model the variation within a class of objects (e.g. hand-written 'A's). For the modeling of the variation during this course, we shall concentrate on statistical pattern recognition, in which the classes and objects within the classes are modeled statistically. For the purposes of this course, we can further divide statistical pattern recognition into two subclasses. Roughly speaking, in one we model the variation within object classes (generative modeling) and in the other we model the variation between the object classes (discriminative modeling). If understood broadly, statistical pattern recognition covers a major part of all pattern recognition applications and systems.

Syntactic pattern recognition forms another class of pattern recognition methods. The basic idea of syntactic pattern recognition is that the patterns (observations about the objects to be classified) can always be represented with the help of simpler and simpler sub patterns leading eventually to atomic

patterns which cannot anymore be decomposed into sub patterns. Pattern recognition is then the study of atomic patterns and the language between relations of these atomic patterns. The theory of formal languages forms the basis of syntactic pattern recognition.

- **Basic Structure of Pattern Recognition Systems**

The task of the pattern recognition system is to classify an object into a correct class based on the measurements about the object. Note that possible classes are usually well-defined already before the design of the pattern recognition system. Many pattern recognition systems can be thought to consist of five stages:

1. Sensing (measurement);
2. Pre-processing and segmentation;
3. Feature extraction;
4. Classification;
5. Post-processing;

Sensing refers to some measurement or observation about the object to be classified. For example, the data can consist of sounds or images and sensing equipment can be a microphone array or a camera. Often one measurement (e.g. image) includes information about several objects to be classified. For instance, assume that we want to recognize the address written on the envelope. We must then classify several characters to recognize the whole address. The data here is probably an image of the envelope including the sub-images of all the characters to be classified and some background that has nothing to do with the pattern recognition task.

Pre-processing refers to filtering the raw data for noise suppression and other operations performed on the raw data to improve its quality. In segmentation, the measurement data is partitioned so that each part represents exactly one object to be classified. For example in address recognition, an image of the whole address needs to be divided to images representing just one character. The result of the segmentation can be represented as a vector that is called a *pattern vector*. Care must be taken during pre-processing because often information is discarded, and if this information is important to the solution of the problem then the overall accuracy of the system can suffer.

Feature extraction - Especially when dealing with pictorial information the amount of data per one object can be huge. A high resolution facial photograph (for face recognition) can contain 1024*1024 pixels. The pattern vectors have then over a million components. The most part of this data is useless for classification. In feature extraction, we are searching for the features that best characterize the data for classification. The result of the feature extraction stage is called a feature vector. The space of all possible feature vectors is called the feature space. In face recognition, a widely used technique to reduce the number features is principal component analysis (PCA). PCA is a statistical technique to reduce the dimensionality of a data vector while retaining most of the information that the data vector contains. In addition to mathematical/computational/statistical techniques, feature extraction can be performed heuristically by picking such features from a pattern vector that could be assumed to be useful in

classification. For face recognition, such a feature can be the distance between the two eyes of a person. Sometimes, dimensionality of the input data is very limited and the pattern vector can be chosen as the feature vector. For example this is the case with the image segmentation methods that are based on the pattern recognition principles. Feature extraction is highly application specific although some general techniques for feature extraction and selection have been developed.

In general, the line between feature extraction and classification is fuzzy. The task of the feature extractor is to produce a representation about the data that enables an easy classification. On the other hand, the task of the classifier is to produce the best classification accuracy given the extracted features. Clearly, these two stages are interdependent from the application point of view. Also, and perhaps more importantly, what is the best representation for the data depends on the classifier applied. There is no such thing as the universally optimal features.

The classifier takes as an input the feature vector extracted from the object to be classified. It places then the feature vector (i.e. the object) to class that is the most appropriate one. In address recognition, the classifier receives the features extracted from the sub-image containing just one character and places it to one of the following classes: 'A','B','C'..., '0','1',...,'9'. The classifier can be thought as a mapping from the feature space to the set of possible classes. Note that the classifier cannot distinguish between two objects with the same feature vector.

Post-processing - A pattern recognition system rarely exists in a vacuum. The final task of the pattern recognition system is to decide upon an action based on the classification result(s). A simple example is a bottle recycling machine, which places bottles and cans to correct boxes for further processing. Different actions can have also different costs associated with them. This information can be included to the classifier design as we will see later on. Also, we can have several interdependent classification results in our hands. For example in an address recognition task, we have the classification results for multiple characters and the task is to decide upon the address that these characters form.

- **Design of Pattern Recognition Systems**

The design of a pattern recognition system is an iterative process. If the system is not good enough, we go back to the drawing table and try to improve some or all of the five stages of the system. After that the system is tested again and it is improved if it still does not meet the requirements placed for it. The process is repeated until the system meets the requirements. The design process is founded on the knowledge about the particular pattern recognition application, the theory of pattern recognition and simply trial and error. Clearly, the smaller the importance of 'trial and error' the faster is the design cycle. This makes it important to minimize the amount of trial and error in the design of the pattern recognition system. The importance of 'trial and error' can be minimized by having good knowledge about the theory of pattern recognition and integrating that with knowledge about the application area. When is a pattern recognition system good enough? This obviously depends on the requirements of the application, but there are some general guidelines about how to evaluate the system. We will return to these later on during this course. How can a system be improved? For example, it can be possible to add a new feature to feature vector to improve the classification accuracy. However, this is not always possible. For

example with the Fisher's Iris dataset, we are restricted to the four existing features because we cannot influence the original measurements. Another possibility to improve a pattern recognition system is to improve the classifier. For this too there exists a variety of different means and it might not be a simple task to pick the best one out of these.

When designing a pattern recognition system, there are other factors that need to be considered besides the classification accuracy. For example, the time spent for classifying an object might be of crucial importance. The cost of the system can be an important factor as well.

- **Supervised and Unsupervised Learning and Classification**

The classifier component of a pattern recognition system has to be taught to identify certain feature vectors to belong to a certain class. The points here are that

1) It is impossible to define the correct classes for all the possible feature vectors and
2) The purpose of the system is to assign an object which it has not seen previously to a correct class.

It is important to distinguish between two types of machine learning when considering the pattern recognition systems. The (main) learning types are supervised learning and unsupervised learning. We also use terms supervised classification and unsupervised classification.

Supervised Classification - In supervised classification [24], we present examples of the correct classification (a feature vector along with its correct class) to teach a classifier. Based on these examples, that are sometimes termed prototypes or training samples, the classifier then learns how to assign an unseen feature vector to a correct class. The generation of the prototypes (i.e. the classification of feature vectors/objects they represent) has to be done manually in most cases. This can mean lot of work: After all, it was because we wanted to avoid the hand-labeling of objects that we decided to design a pattern recognition system in the first place. That is why the number of prototypes is usually very small compared to the number of possible inputs received by the pattern recognition system. Based on these examples we would have to deduct the class of a never seen object. Therefore, the classifier design must be based on the assumptions made about the classification problem in addition to prototypes used to teach the classifier. These assumptions can often be described best in the language of probability theory. To be specific it can be said that a classification procedure is said to be *supervised* if the user

- Defines the decision rules for each class directly or
- Provides training data (class prototypes) for each class to guide the computer classification.

Unsupervised Classification - In unsupervised classification [25] or clustering, there is neither explicit teacher nor training samples. The classification of the feature vectors must be based on similarity between them based on which they are divided into natural groupings. Whether any two feature vectors are similar depends on the application. Obviously, unsupervised classification is a more difficult problem than supervised classification and supervised classification is the preferable option if it is possible. In some cases, however, it is necessary to resort to unsupervised learning. For example, this is the case if the feature vector describing an object can be expected to change with time. The goal in such *unsupervised learning* problems may be to discover groups of similar examples within the data, where it is called *clustering*, or to determine the distribution of data within the input space, known as *density estimation*, or

to project the data from a high-dimensional space down to two or three dimensions for the purpose of *visualization*. To be specific it can be said that a classification procedure is said to be *unsupervised* if

- No training data are required
- The user needs to specify the number of classes

However, there are certain situations where clustering is either useful or necessary. These include:

1. The collection and classification of training data can be costly and time consuming. Therefore, it can be impossible to collect a training set. Also, when there are very many training samples, it can be that all of these cannot be hand-labeled. Then, it is useful to train a supervised classifier with a small portion of training data and use then clustering procedures to fine tune the classifier based on the large, unclassified dataset.
2. For data mining, it can be useful to search for natural clusters/groupings among the data, and then recognize the clusters.
3. The properties of feature vectors can change over time. Then, supervised classification is not reasonable, because sooner or later the test feature vectors would have completely different properties than the training data had. For example, this problem is commonplace in the processing of medical images.
4. The clustering can be useful when searching for good parametric families for the class conditional densities for the supervised classification.

Above situations only are examples of the situations where clustering is worth-while. There are many more situations requiring clustering.

In my PhD work both supervised and unsupervised classification techniques have been used. I have used *Neural Network (NN), Support Vector Machine (SVM)* and *RANdom SAmple Consensus (RANSAC)* as example of supervised classification technique. I have used *k-means clustering* as example of unsupervised classification technique. Each of this classifier is described below-

1) **Neural Network** – Artificial neural networks are algorithms which have been developed to tackle a range of computational problems. Artificial neural networks were originally introduced as very simplified models of brain function, so it is initially instructive to consider the broad analogies between artificial neural networks and biological ones. Of course, the latter are considerably more complex, and many artificial neural network models today bear very little resemblance to biological ones.

The human brain consists of billions of interconnected neurons. These are cells which have specialized membranes which allow the transmission of signals to neighboring neurons. A single axon extends from the basic cell body. This is the output from the neuron, and typically divides into many sub-branches before terminating at a synapse. Electrical pulses are transmitted along the axon to the synapses by the transfer of Na+ ions. The arrival of a voltage pulse stimulates the release of neurotransmitting chemicals across the synaptic cleft towards the postsynaptic cell, which is the receiving part of the next neuron. This postsynaptic cell passes the signal via the dendrite to the main part of the neuron body. The inputs from the different dendrites are then combined to produce an output signal which is passed along the axon, and so the process continues. However, a signal is only produced in the axon if there are enough inputs of

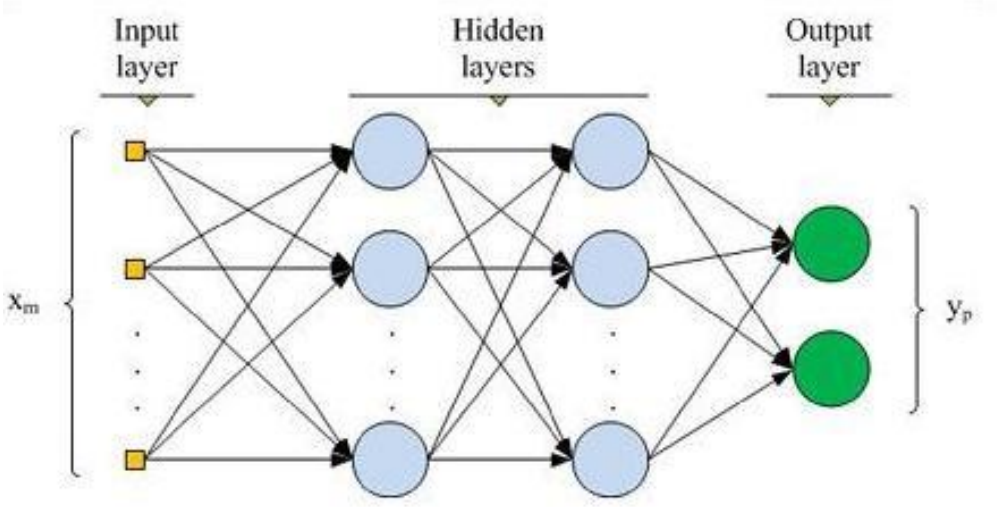
sufficient strength to overcome some threshold value, and then the output is some nonlinear function of the input stimuli.

The human brain consists of about 10^{11} neurons, and each neuron has between a few and a few thousand synapses on its dendrites, giving a total of about 10^{14} synapses (connections) in the brain. The 'strength' of the synaptic connection between neurons can be chemically altered by the brain in response to favorable and unfavorable stimuli, in such a way as to adapt the organism to function optimally within its environment. The synapses are therefore believed to be the key to learning in biological systems.

An artificial neural network (ANN), usually called neural network (NN), is a mathematical model or computational model that is inspired by the structure and/or functional aspects of biological neural networks. "Artificial neural networks" is a relatively loose term referring to mathematical models which have some kind of distributed architecture, that is, consist of processing nodes (analogous to neurons) with multiple connections (analogous to dendrites and axons). These connections generally have adaptable parameters which modify the signals which pass along them. There are numerous types of artificial neural networks for addressing many different types of problems, such as modeling memory, performing pattern recognition, and predicting the evolution of dynamical systems. Most networks therefore perform some kind of data modeling.

A neural network consists of an interconnected group of artificial neurons, and it processes information using a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase. Modern neural networks are non-linear statistical data modeling tools. They are usually used to model complex relationships between inputs and outputs or to find patterns in data.

The word network in the term 'artificial neural network' refers to the inter–connections between the neurons in the different layers of each system. An example system has three layers. The first layer has input neurons, which send data via synapses to the second layer of neurons, and then via more synapses to the third layer of output neurons. The synapses store parameters called "weights" that manipulate the data in the calculations.



Basic architecture of the MLP

An ANN is typically defined by three types of parameters:

1. The interconnection pattern between the different layers of neurons
2. The learning process for updating the weights of the interconnections
3. The activation function that converts a neuron's weighted input to its output activation.

The simplest neural network is the perceptron. The complex network topologies are mainly categorized as:

- ✓ **recurrent**: arbitrary connections between cells in different layers, enabling feedback loops. A **recurrent neural network** (RNN) [21, 23] is any network whose neurons send feedback signals to each other.
- ✓ **non-recurrent**: recurrent connectivity is not permitted, i.e., information flows in forward direction (from input to output).
- ✓ **feedforward**: a special type of non-recurrent network where forward connectivity is allowed between neighboring layers. The feedforward neural network was the first and simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network.

The feedforward, back-propagation network is the most popular, which is known as multilayer perceptron (MLP) [35]. The term perceptron is historical, and refers to the function performed by the nodes. Feedforward means that there is a definite input and output, and a flow of data in one direction. This is in contrast to recurrent neural networks in which data flows in a loop: these are important when time plays a relevant role in the problem.

The backpropagation network has one input layer, one output layer and at least one hidden layer. Each node in a layer is connected to all other nodes in the next layer. The number of layers and number of cells per layer are important parameters in the design of an ANN. But, there is no clear guideline to the optimal choice of these parameters for a particular application. Deciding upon the appropriate number of hidden nodes and layers is largely a matter of experience. The basic theory of backpropagation is simple, a number of tricks — some a bit subtle — are often used to improve performance and increase training speed. Choices involving the scaling of input values and initial weights, desired output values, and more can be made based on an analysis of networks and their function.

Network architecture or topology plays an important role for neural net classification, and the optimal topology will depend upon the problem at hand. It is here that another great benefit of networks becomes apparent: often knowledge of the problem domain which might be of an informal or heuristic nature can be easily incorporated into network architectures through choices in the number of hidden layers, units, feedback connections, and so on. Thus setting the topology of the network is heuristic model selection. The practical ease in selecting models (network topologies) and estimating parameters (training via backpropagation) enable classifier designers to tryout alternate models fairly simply. In this process, one input is feed to the network and the system learns by comparing the output value and target value. The error quantity is fed back into the network to modify the weights. Repeat the process with another data from the training set.

- **Advantages**

1) Data driven and self adaptive.
2) Flexible for real world applications.
3) High accuracy and noise tolerance.

- **Disadvantages**

1) Lack of transparency.
2) Learning time is long.
3) Defining classification rules is difficult.

2) **Support Vector Machine (SVM)** – Improving classifier effectiveness has been an area of intensive machine-learning research over the last two decades, and this work has led to a new generation of state-of-the-art classifiers, such as support vector machines. Kernel-based techniques (such as support vector machines, Bayes point machines, kernel principal component analysis, and Gaussian processes) represent a major development in machine learning algorithms. Support vector machines (SVM) are a group of supervised learning methods that can be applied to classification or regression. Support vector machines represent an extension to nonlinear models of the generalized portrait algorithm developed by Vladimir Vapnik. The SVM algorithm is based on the statistical learning theory and the Vapnik-Chervonenkis (VC) dimension introduced by Vladimir Vapnik and Alexey Chervonenkis [30]. Support Vector Machine is of two types – Linear nd Non-linear. In linear Support Vector Machine two classes are separated, but in non-linear Support Vector Machine classification of multiple class is possible. Linear Support Vector Machine is comparatively much more simpler than non-linear Support Vector Machine.

A support vector machine (SVM) analyzes data and recognizes patterns. It is used for classification and regression analysis. The standard SVM takes a set of input vectors and transform them into high-dimensional space using linear or nonlinear transformation, and then execute a linear separation in the higher dimensional space. The basic SVM is a linear binary classifier. The classifier is trained on a set of two-class linearly separable vectors.

Let the training data is $\{(x_i, y_i) : 1 \leq i \leq p\}$ where the observation (i.e., feature vector) $x_i \in R^n$, class $y_i \in \{-1,+1\}$ and p is the size of the data set. Let the data set is separable by the hyperplane H_0 which is denoted as

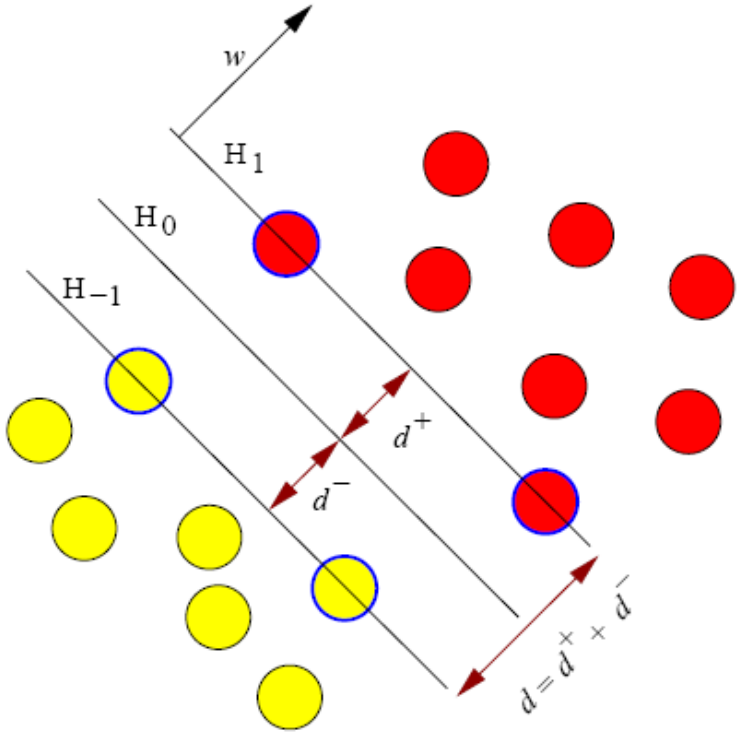
$$H_0 : w^t x + b = 0$$

where w is normal to H_0, t represents the transpose, x is a point on H_0 and $b \in R$. The data space is partitioned into two subspaces, positive (respectively, negative) if $w^t x+b > 0$ (< 0, respectively). Let, d^+ (d^- , respectively) be the shortest distance of the closest point from positive (negative, respectively) subspace to H_0. The distance $d = d^+ + d^-$ is defined as the margin of H_0 and basic target to maximize the quantity d. The constraints for this optimization problem can be formulated as

$$w^t x_i + b \geq 1 \text{ for } y_i = 1$$
$$w^t x_i + b \leq -1 \text{ for } y_i = -1$$

The above constraints can be combined as

$$y_i(w^t x_i + b - 1) \geq 0 \; \forall i$$



Two class linear classification: support vectors are encircled by blue boundary

Using Lagrange technique, the optimal solution w can be found as

$$w = \sum_{i=1}^{p} \alpha_i y_i x_i$$

where α_is are Lagrange multipliers. The points x_i with _i > 0 and lying on the hyperplane $H_1 : w^t x_i + b = 1$ or $H_1 : w^t x_i + b = 1$ are known as the support vectors. After the training part of the SVM is completed, the class of a test pattern x, i.e., c(x), is determined by considering sign of $w^t x + b$, i.e.,

$$c(x) = \text{sgn}\ [w^t x + b] = \text{sgn}[\sum_{i=1}^{p} \alpha_i y_i x_i^t x + b]$$

where sgn[.] is the sign function. The product $x^t i\ x$ of equn represents the linear characteristics of the SVM. If the distribution of the data set is not linearly separable, but non-linearly separable, then SVM can be constructed using a kernel function $K(x_i, x)$ in place of $x^t_i x$ of equn. In this case, the class of a test pattern x can be determined as

$$c(x) = \text{sgn}[\sum_{i=1}^{p} \alpha_i y_i K(x_i, x) + b]$$

where the model parameter w and b determine through training. Commonly used kernel functions in SVM applications are:

$$K(\mathbf{X}_i, \mathbf{X}_j) = \begin{cases} \mathbf{X}_i \cdot \mathbf{X}_j & \text{Linear} \\ (\gamma \mathbf{X}_i \cdot \mathbf{X}_j + C)^d & \text{Polynomial} \\ \exp(-\gamma |\mathbf{X}_i - \mathbf{X}_j|^2) & \text{RBF} \\ \tanh(\gamma \mathbf{X}_i \cdot \mathbf{X}_j + C) & \text{Sigmoid} \end{cases}$$

where,

$$K(\mathbf{X}_i, \mathbf{X}_j) = \phi(\mathbf{X}_i) \bullet \phi(\mathbf{X}_j)$$

that is, the kernel function, represents a dot product of input data points mapped into the higher dimensional feature space by transformation θ. The kernel θ is used to transform data from the input (independent) to the feature space.

The SVM is binary classier in nature, it can also be used to handle multiclass problem. The simplest approach is "one-versus-rest" to train N classiers and determine the class of a test data.

The *advantages* of support vector machines are:

- Effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Versatile: different *Kernel functions* can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The *disadvantages* of support vector machines include:

- If the number of features is much greater than the number of samples, the method is likely to give poor performances.
- Difficulty in choosing the kernel, the selection of kernel function parameters
- Both training and testing speed
- Algorithmic complexity
- Memory requirement

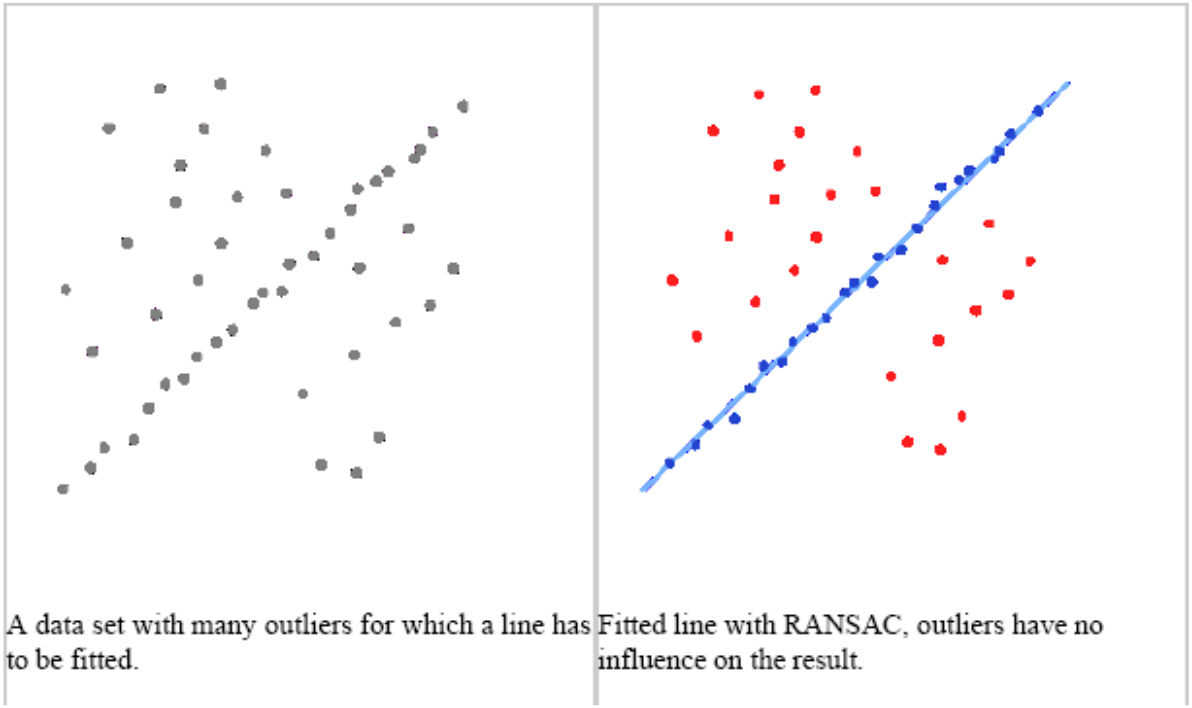
3) **RANdom SAmple Consensus (RANSAC)** – The algorithm was first published by Fischler and Bolles in 1981 [4]. RANSAC is an abbreviation for "RANdom SAmple Consensus". It is an iterative method to estimate parameters of a mathematical model from a set of data contaminated by large number of outliers (that do not fit the model). The major strength of RANSAC over other estimators lies in the fact that the estimation is made based on inliers i.e. whose distribution can be explained by a set of model parameters. It can produce reasonably good model provided a data set contains a sizable amount of inliers. It may be noted that RANSAC can work satisfactorily even with outliers amounting to 50% of entire data set.

RANSAC algorithm is primarily composed of two steps – hypothesize and test. These are executed iteratively. During hypothesize phase, a minimal sample set is randomly selected and model parameters are computed based on only the elements in the selected sample set. In the test phase, consistency of all

the elements in the entire data set are verified to check whether or not they are consistent with the model obtained. Consistent elements are included to form the new sample set. The process goes on iteratively and in each iteration a model is obtained. Finally, the model that fits best is taken as the estimate.

RANSAC achieves its goal by iteratively selecting a random subset of the original data. These data are *hypothetical inliers* and this hypothesis is then tested as follows:

1. A model is fitted to the sample of hypothetical inliers, i.e. all free parameters of the model are fitted from this sample.
2. All other data are then tested against the fitted model and, those points that fit the estimated model well are considered as part of the *consensus set*.
3. The estimated model is reasonably good if sufficiently many points have been classified as part of the consensus set.
4. Afterwards, the model may be improved by reestimating it using all members of the consensus set.



A data set with many outliers for which a line has to be fitted.

Fitted line with RANSAC, outliers have no influence on the result.

- **Assumptions**

I. An all-inlier sample generates model consistent with all inliers.
II. A model consistent with a sample contaminated by at least one outlier has small support.
III. All-inlier sample, even in the presence of noise, gives a model that is close to the optimal one. A local optimization of the model parameters reaches the optimal model that is supported by all inliers.

Considering data elements to be n-dimensional, RANSAC tries to fit a hyperplane and estimates the model parameters. Let the hyperplane is represented as

$$w_0 + w_1d_1 + w_2d_2 + \ldots + w_nd_n = 0$$

where, $< d_1, d_2, \ldots d_n >$ be the n-dimensional point. It estimates the value of wis by minimizing the fitting error for each element in the entire data set. An element e_i is considered as an inlier/consistent with respect to the model provided its orthogonal regression, $d(e_i,W)$ with the model is within the threshold δ; where,

$$d(e_i, W) = \frac{w_0 + \sum_j w_j e_{ij}}{\sqrt{w_0^2 + \sum_j w_j^2}}$$

and ei $=< e_{i1}, e_{i2} \ldots e_{in} >$. In our experiment, δ is taken as 0.02 as suggested in. $\sum d(e_i,W)$ is taken as the total fitting error for the model under consideration. The model with minimum error is considered as the final one.

Classically, RANSAC is an estimator for the parameters of a model from a given data set. But, in this work, it has been used a classifier. Corresponding to the data set of each category a model is estimated first. Then for a given element, its class can be determined easily by finding the best matched model. As it has been discussed that RANSAC estimates the model relying on the inliers, unlike other technique, it is less affected by the noisy data. Thus, RANSAC is well suited for my purpose.

- **Algorithm**

Step 1: Select randomly the minimum number of points required to determine the model parameters
Step 2: Solve for the parameters of the model
Step 3: Determine how many points from the set of all points fit with a predefined tolerance ε
Step 4: If the fraction of the number of inliers over the total number points in the set exceeds a predefined threshold τ, re-estimate the model parameters using all the identified inliers and terminate
Step 5: Otherwise, repeat steps 1 through 4 (maximum N times)

- **Advantages and disadvantages**

An advantage of RANSAC is its ability to do robust estimation of the model parameters, i.e., it can estimate the parameters with a high degree of accuracy even when a significant number of outliers are present in the data set. A disadvantage of RANSAC is that there is no upper bound on the time it takes to compute these parameters. When the number of iterations computed is limited the solution obtained may not be optimal, and it may not even be one that fits the data in a good way. In this way RANSAC offers a trade-off; by computing a greater number of iterations the probability of a reasonable model being produced is increased. Moreover, RANSAC is not always able to find the optimal set even for moderately contaminated sets and it usually performs badly when the number of inliers is less than 50%. Optimal RANSAC was proposed to handle both these problems and is capable of finding the optimal set for

heavily contaminated sets, even for an inlier ratio under 5%. Another disadvantage of RANSAC is that it requires the setting of problem-specific thresholds.

RANSAC can only estimate one model for a particular data set. As for any one-model approach when two (or more) model instances exist, RANSAC may fail to find either one. The Hough transform is one alternative robust estimation technique that may be useful when more than one model instance is present. Another approach for multi model fitting is known as PEARL, which combines model sampling from data points as in RANSAC with iterative re-estimation of inliers and the multi-model fitting being formulated as an optimization problem with a global energy functional describing the quality of the overall solution.

4) **k-means clustering** – **Cluster analysis** or **clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense or another) to each other than to those in other groups (clusters). It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.

Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions. Clustering can therefore be formulated as a multi-objective optimization problem. The appropriate clustering algorithm and parameter settings (including values such as the distance function to use, a density threshold or the number of expected clusters) depend on the individual data set and intended use of the results. Cluster analysis as such is not an automatic task, but an iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It will often be necessary to modify data preprocessing and model parameters until the result achieves the desired properties.

Cluster analysis was originated in anthropology by Driver and Kroeber in 1932 and introduced to psychology by Zubin in 1938 [28] and Robert Tryon in 1939 [29] and famously used by Cattell beginning in 1943 for trait theory classification in personality psychology.

- **Centroid-based clustering**

In centroid-based clustering, clusters are represented by a central vector, which may not necessarily be a member of the data set. When the number of clusters is fixed to k, ***k*-means clustering** gives a formal definition as an optimization problem: find the $k$cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized.

Most k-means-type algorithms require the number of clusters - k- to be specified in advance, which is considered to be one of the biggest drawbacks of these algorithms. Furthermore, the algorithms prefer clusters of approximately similar size, as they will always assign an object to the nearest centroid. This often leads to incorrectly cut borders in between of clusters (which are not surprising, as the algorithm optimized cluster centers, not cluster borders).

Clustering techniques have been applied to a wide variety of research problems. Hartigan (1979) [27] provides an excellent summary of the many published studies reporting the results of cluster analyses. For example, in the field of medicine, clustering diseases, cures for diseases, or symptoms of diseases can

lead to very useful taxonomies. In the field of psychiatry, the correct diagnosis of clusters of symptoms such as paranoia, schizophrenia, etc. is essential for successful therapy. In archeology, researchers have attempted to establish taxonomies of stone tools, funeral objects, etc. by applying cluster analytic techniques. In general, whenever we need to classify a "mountain" of information into manageable meaningful piles, cluster analysis is of great utility.

k-means [26] is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters). The main idea is to define k centroids, one for each cluster. These centroids should be placed in such a way that they are as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early grouping is done. At this point we need to re-calculate k new centroids by taking the mean value of all of the samples assigned to each previous centroid. The difference between the old and the new centroids are computed and the algorithm repeats these last two steps until this value is less than a threshold. In other words, it repeats until the centroids do not move significantly.

Finally, this algorithm aims at minimizing an *objective function*, in this case a squared error function. The objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

where, $\left\| x_i^{(j)} - c_j \right\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, is an indicator of the distance of the *n* data points from their respective cluster centers.

- **Algorithm 1 (K-means clustering)**

1 **begin initialize** *n, c,*μ_1,μ_2, . . . ,μ_c
2 **do** classify *n* samples according to nearest μ_i
3 recompute μ_i
4 **until** no change in μ_i
5 **return** μ_1,μ_2, . . . ,μ_c
6 **end**

The computational complexity of the algorithm is O ($ndcT$) where d the number of features and T the number of iterations, c is the number of clusters and n is the number of samples. In practice, the number of iterations is generally much less than the number of samples. The values obtained can be accepted as the answer, or can be used as starting points for the more exact computations. The algorithm has converged when the assignments no longer change.

- **Properties**

1) There are always K clusters.
2) There is always at least one item in each cluster.
3) The clusters are non-hierarchical and they do not overlap.
4) Every member of a cluster is closer to its cluster than any other

- **Advantages**

1) Fast, robust and easier to understand.
2) Relatively efficient.
3) Gives best result when data set are distinct or well separated from each other.

- **Disadvantages**

1) The learning algorithm requires specification of the number of cluster centers.
2) The use of Exclusive Assignment - If there are two highly overlapping data then k-means will not be able to resolve that there are two clusters.
3) Randomly choosing of the cluster center cannot lead us to the fruitful result.
4) Unable to handle noisy data and outliers.
5) Algorithm fails for non-linear data set.
6) Different initial partitions can result in different final clusters.

References

[1] Shao, Yang, and Ross S. Lunetta. "Comparison of support vector machine, neural network, and CART algorithms for the land-cover classification using limited training data points." *ISPRS Journal of Photogrammetry and Remote Sensing* 70 (2012): 78-87.

[2] Freitas, Cinthia OA, et al. "Metaclasses and zoning mechanism applied to handwriting recognition." *J Univers Comput Sci* 14.2 (2008): 211-223.

[3] http://en.wikipedia.org/wiki/RANSAC

[4] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model for model fitting with applications to image analysis and automated cartography," *ACM Communications*, vol. 24, pp. 381–395, 1981.

[5] Zhang, Tong, and C-C. Jay Kuo. "Content-based classification and retrieval of audio." *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*. International Society for Optics and Photonics, 1998.

[6] Li, Tao, Mitsunori Ogihara, and Qi Li. "A comparative study on content-based music genre classification." *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 2003.

[7] Jiang, Dan-Ning, et al. "Music type classification by spectral contrast feature." *Multimedia and Expo, 2002. ICME'02. Proceedings. 2002 IEEE International Conference on*. Vol. 1. IEEE, 2002.

[8] Foote, Jonathan T. "Content-based retrieval of music and audio." *Voice, Video, and Data Communications*. International Society for Optics and Photonics, 1997.
[9] John A. Bullinaria, Lecture Notes on "Recurrent Neural Networks" Neural Computation : Lecture 12
[10] NPTEL Lecture Notes
[11] F. Rosenblatt, "Principles of Neurodynamics: Perceptrons and the theory of Brain Mechanisms", *Spartan Books*, Washington DC, 1962.
[12] Cover T. M. and Hart P. E., "Nearest Neighbor Pattern Classification", *IEEE Transactions on Informnation Theory*, Vol. 13, pp. 21-27, 1967
[13] R. Xu and D. C. Wunsch, II, "Clustering", *IEEE Press*, 2009.
[14] Bishop C. M., "Pattern Recognition and Machine Learning", *Springer*, 2006
[15] Duda, R. O.,Hart P. E., and Stork D. G., "Pattern Classification", *John Wiley & Sons*, 2004
[16] K-means and Hierarchical Clustering Tutorial Slides by Andrew Moore
[17] Zheng, Lihong, and Xiangjian He. "Classification Techniques In Pattern Recognition." (2005).
[18] http://www.mathworks.in
[19] Jain, A.K; Murty M. N; Flynn, P. J. " Data Clustering: A Review", ACM Computing Surveys, Vol. 31, No. 3, pp 264-323, 1999
[20] Devi, V.S; Murty M. N. "Pattern Recognition: An Introduction", Universities Press,Hyderabad, 2011
[21] Graves, Alex, et al. "A novel connectionist system for unconstrained handwriting recognition." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 31.5 (2009): 855-868.
[22] R. Plamondon and S. N. Srihari, "On-line and off-line handwriting recognition: A comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 63–84, 2000.
[23] Hüsken, Michael, and Peter Stagge. "Recurrent neural networks for time series classification." *Neurocomputing* 50 (2003): 223-235.
[24] K. Doya. Recurrent networks: Supervised learning. In M. A. Arbib, editor, The Handbook of Brain Theory and Neural Networks, pages 796-800. The MIT Press, Cambridge, MA, 1998.
[25] Shao, Xi, Changsheng Xu, and Mohan S. Kankanhalli. "Unsupervised classification of music genre using hidden markov model." *Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on*. Vol. 3. IEEE, 2004.
[26] Kanungo, Tapas, et al. "An efficient k-means clustering algorithm: Analysis and implementation." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24.7 (2002): 881-892.
[27] Hartigan, John A., and Manchek A. Wong. "Algorithm AS 136: A k-means Clustering Algorithm." *Applied statistics* (1979): 100-108.
[28] Zubin, Joseph. "A technique for measuring like-mindedness." *The Journal of Abnormal and Social Psychology* 33.4 (1938): 508.
[29] Tryon, Robert Choate. *Cluster analysis: correlation profile and orthometric (factor) analysis for the isolation of unities in mind and personality*. Edwards brothers, Incorporated, lithoprinters and publishers, 1939.
[30] Cortes, Corinna, and Vladimir Vapnik. "Support Vector Machine." *Machine learning* 20.3 (1995): 273-297.
[31] Xuegong, Zhang. "Introduction to statistical learning theory and support vector machines." *Acta Automatica Sinica* 26.1 (2000): 32-42.

[32] Joachims, Thorsten. *Text categorization with support vector machines: Learning with many relevant features*. Springer Berlin Heidelberg, 1998.
[33] Shawe-Taylor, C. A., and Smola Schölkopf. "The Support Vector Machine." (2000).
[34] Zhang, Li, Weida Zhou, and Licheng Jiao. "Wavelet support vector machine." *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 34.1 (2004): 34-39.
[35] Gardner, M. W., and S. R. Dorling. "Artificial neural networks (the multilayer perceptron)--a review of applications in the atmospheric sciences." *Atmospheric environment* 32.14-15 (1998): 2627-2636.
[36] Ruck, Dennis W., Steven K. Rogers, and Matthew Kabrisky. "Feature Selection Using a Multilayer Perceptron." *Journal of Neural Network Computing* 2.2 (1990): 40-48.
[37] Derpanis, Konstantinos G. "Overview of the RANSAC Algorithm." *York University* 68 (2010).