IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 20, Issue 3, Ver. II (May. - June. 2018), PP 06-13
www.iosrjournals.org

Rainfall Prediction Using Support Vector Machine (SVM)

Jyoti Ranjan Mohanty[1], Manas Ranjan Mohapatra[2]
[1]*Department of Computer Science andApplication, OUAT,India*
[2]*Department of Computer Science, Banki College (Autonomous), India*
Corresponding Author:Jyoti Ranjan Mohanty

Abstract:*Rainfall is considered to be an important ingredient in agricultural cultivation. Conventional rain fall predictors use ANN to predict the future rainfall. We introduce a simple machine learning based predictive model using SVM regression. Here the Average Mean Square Error (AMSE) for each kernel is evaluated and the kernel having minimum MSE is selected for prediction. The proposed technique is applied to predict the month wise prediction of rainfall in Khurda District of Orissa. After successful training and validation, the result derive from SVM model is Polynomial kernel is low MSE among all. But after changing the parameter value for each kernel with specified run it is found the Linear kernel produce minimum average MSE 15.04% on test data set while other kernels like Polynomial, RBF, Sigmoid produce 18.81%, 16.15% & 18.32% respectively.*

Keywords *-Support Vector Machine, Kernel function, RBF kernel, Polynomial Kernel, Accuracy*

Date of Submission: 18-05-2018 Date of acceptance: 02-06-2018

I. Introduction

Agriculture is the predominant occupation in Orissa, accounting for about 52% of employment. The Irrigation facilities are inadequate, as revealed by the fact that only 52.6% of the land was irrigated in 2009– 10 which result in farmers still being dependent on rainfall, specifically the Monsoon season. A good monsoon results in a robust growth for the economy as a whole, while a poor monsoon leads to a sluggish growth. So, in the last few decades or so, machine learning techniques such as Artificial Neural Networks (ANN)[1], fuzzy logic, genetic programming, etc., have been widely used in the modelling and prediction of hydrologic variables used in rainfall. One common way to improve the prediction accuracy is to perform some pre-processing of the inputs. Changing the representation of data is one such technique, for example. Ideally, a pre-processing that most matches the specific learning problem should be chosen. In this study, Rain Fall Analysis is proposed as a novel processing technique for the deterministic chaotic systems, e.g. the rainfall prediction processes, and the resulting input representation is trained with Support Vector Machine (SVM)[3] for forecasting. Although it adopt noise-reduction (filtering) technique[2], in this study SVM used an efficient pre-processing algorithm which results in the modified representation of the input vectors where new features are linear functions of the original attributes. This is because to provide for Thus, the prediction accuracy may be better when the learning machine is presented with all components of the analysis for training. However, such an approach has an obvious disadvantage in terms of the cost one has to pay for the computational and generalization performance of the learning machine, which degrades rapidly with the growth in the number of input features.

In this work a SVM is proposed to overcome this problem. SVM offers an efficient way to deal with the computational and generalization performance in a high-dimensional input space owing to the dual representation of the machine in which the training patterns always appear in the form of scalar products between pairs of examples. In summary, this paper addresses the forecasting problem in two steps:

1. Pre-processing the input based on data set & represent it into a set of high and low data which result in a high dimensional input space.
2. Training the Support Vector Machine (SVM) to learn this preprocessed data and subsequent prediction.

II. Related Work

Vautard *et al*. (1992)[4], is generally used to perform a spectrum analysis on the input data, eliminate the 'irrelevant features' (high-frequency components) and invert the remaining components to yield a 'filtered' time series. This approach of filtering a time series to retain desired modes of variability is based on the idea that the predictability of a system can be improved by forecasting the important oscillations in time series taken from the system.

The general idea is to filter the record first and then use some model to forecast on the filtered series (Elsner & Tsonis, 1997). For example, Lisi *et al*. (1995)[5] applied SSA to extract the significant components in

their study on Southern Oscillation Index (SOI) time series and used a back-propagation neural network for prediction. They reconstructed the original series by summing up the first '*p*' significant components.

Deterministic chaotic systems like the rainfall and runoff processes (Jayawardena & Lai, 1994[7]; Sharifi *et al*., 1990[8]; Sivakumar *et al*. [9], 1998; Islam *et al*., 2000[10]), it is difficult to precisely demarcate signal and noise components and the suppression of certain high frequency components may alter the resulting filtered output signal.

Mukherjee *et al*. (1997)[11] applied SVM for non-linear prediction of chaotic time series (the Mackey-Glass time series, the Ikeda map and the Lorenz time series) and compared the results with different approximation techniques (ANN, polynomial, RBFs, local polynomial and rational).

Dibike (2000)[12] concluded that SVM does generalize better than both ANN and genetic programming in his case study of rainfallrunoff modeling.

Babovic *et al*. (2000)[13] concluded that SVM produced consistently better results over 12 lead periods than ANN for water level forecasting in the city of Venice.

Liong & Sivapragasam (2000)[14] and Sivapragasam & Liong (2000) demonstrated that SVM shows good generalization performance in their applications on flood forecasting and rainfall modeling, respectively.

III. Support Vector Machines(SVM)

The foundations of Support Vector Machines (SVM) have been developed by Vapnik and gained popularity due to many promising features such as better empirical performance. The formulation uses the Structural Risk Minimization (SRM) principle, which has been shown to be superior to traditional Empirical Risk Minimization (ERM) principle, used by conventional neural networks. SRM minimizes an upper bound on the expected risk, where as ERM minimizes the error on the training data. It is this difference which equips SVM with a greater ability to generalize, which is the goal in statistical learning.

3.1 Statistical Learning Theory

In statistical learning theory the problem of supervised learning is formulated as follows. We are given a set of training data {(**x**1,y1)... (**x**l,yl)} in Rn ´ R sampled according to unknown probability distribution P(**x**,y), and a loss function V(y,f(**x**)) that measures the error, for a given **x**, f(**x**) is "predicted" instead of the actual value y. The problem consists in finding a function f that minimizes the expectation of the error on new data that is, finding a function f that minimizes the expected error given in equation (1):

$$\int V(y,f(x))P(x,y)dx\,dy \qquad (1)$$

In statistical modelling we would choose a model from the hypothesis space, which is closest (with respect to some error measure) to the underlying function in the target space.

SVM and SVR are a set of related supervised learning methods used for classification and regression respectively. They belong to a family of generalised linear classifiers. A special property of SVMs is that they simultaneously minimise the empirical classification error and maximise the geometric margin; hence they are also known as maximum margin classifiers.

3.2 Learning and Generalization

Early machine learning algorithms aimed to learn representations of simple functions. Hence, the goal of learning was to output a hypothesis that performed the correct classification of the training data and early learning algorithms were designed to find such an accurate fit to the data. The ability of a hypothesis to correctly classify data not in the training set is known as its generalization. SVM performs better in term of not over generalization when the neural networks might end up over generalizing easily. Another thing to observe is to find where to make the best trade-off in trading complexity with the number of epoch which shown in theFig1

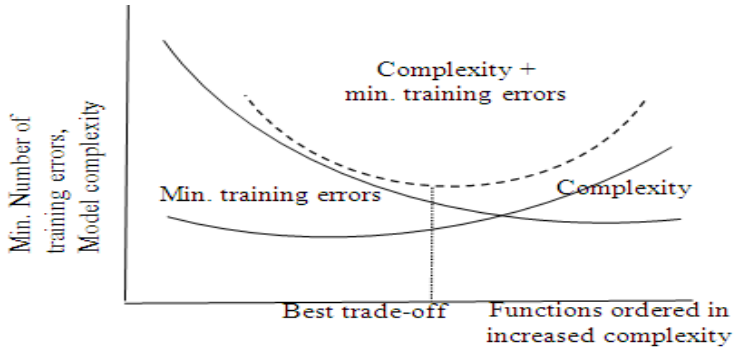

Fig. 1. Number of Epochs Vs Complexity

3.3 Types of SVM

The simplest Support Vector Machine is linear SVM which uses linear decision boundary. But in case of non-separable data set, linear SVM is not very effective in classification. Thus, we use non-linear decision boundary to classify non-separable data sets. As Non-linear SVM is an extension of liner SVM, a very brief overview of linear SVM is given followed by non-linear SVM.

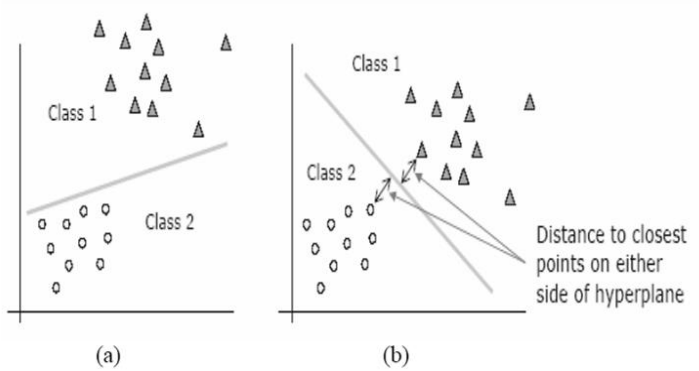

Fig. 2. Classification of two classes by hyperplane

In the Fig. 2, there are two classes, circles (representing desired value -1) and triangles representing desired value 1). Intuitively, we feel that the hyperplane (in 2 dimension, hyperplane is a line) in the Fig. 1(b) classify the two classes better than the other hyperplane shown. This is because the hyperplane in 1(b) is equally distant from the closest data points of the two classes and will give good generalization result for unseen data points. Support Vector Machine is based on this idea.

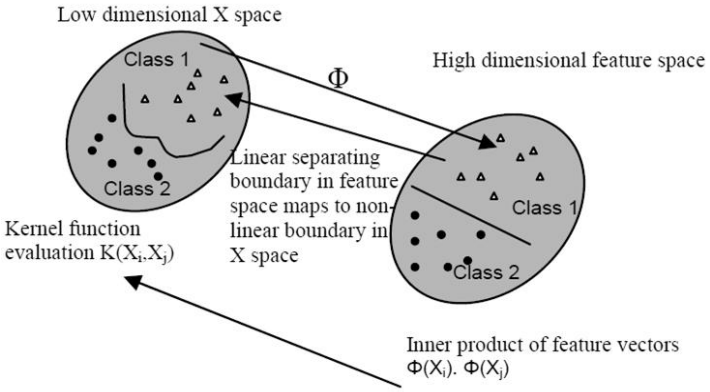

Fig. 3. Non-Linear separation & Kernel Function

In the Linear SVM, a linear decision boundary is used to classify the training set. In case of non-separable data, the data set is not completely classified by linear decision boundary. By using non-linear decision boundary, non-separable points can be classified correctly. In general, models are not scalable from linear region to non-linear region but SVM can be converted from linear to non-linear mode with few changes. Non-linear SVM employs a non-linear decision function to classify the training set by mapping the non-separable data points to higher dimension where these data points become separable. In the higher dimension, a linear decision boundary is located to classify the data set. This linear decision boundary becomes non-linear boundary when mapped back into input space.

In Non-Linear SVM Kernel function chosen results in different kinds of problems with different performance levels, and the choice of the appropriate Kernel for a specific application is often a difficult task. A necessary and sufficient condition for a Kernel to be valid, but other than that, there is really no mathematically structured approach to prefer one kernel over the other. An obvious choice, though, is that if the data is known to be not linearly separable, we would expect that a non-linear kernel would perform better than the one based on a linear kernel.

However it is often the case that the data is far from linear and the datasets are inseparable. To allow for this kernels are used to non-linearly map the input data to a high-dimensional space. This mapping is defined by the Kernel in equation(2):

$K(x_i,x_j) = \Phi(x_1).\ \Phi(x_2)$ (2)

IV. Sum Kernels

Here we see that we need to represent the dot product of the data vectors used. The dot product of nonlinearly mapped data can be expensive. The kernel trick just picks a suitable function that corresponds to dot product of some nonlinear mapping instead. Some of the most commonly chosen kernel functions are given below in later part of this tutorial. A particular kernel is only chosen by trial and error on the test set, choosing the right kernel based on the problem or application would enhance SVM' s performance.
The following are the kernels investigated in this project:

4.1 Linear Kernel

The kernel function is defined as

$$K(x_i, x_j) = 1 + x_i^T x_j$$

This is the simplest kernel and shows good performance for linearly separable data. Surprisingly, works very well even in cases of non-linear data.

4.2 Polynomial Kernel

The kernel function is defined as

$$K(x_i, x_j) = (1 + x_i^T x_j)^p$$

Where*p* is the degree of the polynomial. The motivation is that in general, for vectors $\mathbf{x}_i$ that are linearly dependent on *p* dimensions, the kernel function of order *p* can be used to transform them into linearly independent vectors on those *p* dimensions. Once they are transformed into the dimension space where they become linearly separable, the linear-SVM case can handle the classification problem. Thus, in a way, it is an extension of the linear kernel, in that it gives the crucial transformation to " enable independence" among the training samples. The performance of this kernel is expected to be around the same as that of the linear kernel, since the principle behind the two is the same and the transformation is to just take them to different space. However, the performance does depend on the order *p* of the polynomial, since how well the data becomes separable depends on it.

4.3 Radial Basis Function (RBF) Kernel

The kernel function is defined as

$$K(x_i, x_j) = \exp(-\frac{|| x_i - x_j ||^2}{2\sigma^2})$$

This kernel is basically suited best to deal with data that have a class-conditional probability distribution function approaching the Gaussian distribution. It maps such data into a different space where the data becomes linearly separable. To actually visualize this, it is convenient to observe that the kernel (which is exponential in nature) can be expanded into an infinite series, thus giving rise to an infinite-dimension polynomial kernel: each of these polynomial kernels will be able to transform certain dimensions to make them linearly separable.

Naturally, one would expect the RBF kernel to perform much better than either the Linear or the Polynomial kernel. However, this kernel is difficult to design, in the sense that it is difficult to arrive at an optimum σ and choose the corresponding *C* that works best for a given problem. The fact that certain combinations of σ and *C* make the SVM highly sensitive to training data also contributes to the error rate of the RBF-based SVM.

One of the advantages of the RBF kernel is that given the kernel, the weights α_i, the number of support vectors Ns and the support vectors si are all automatically obtained as part of the training procedure, i.e, they need not be specified by the training mechanism. Thus, it is invulnerable to this particular subset of design issues, while it does remain highly vulnerable to the other design issue, namely the kernel to be used and the cost to associate.

4.4 Sigmoid Kernel

The kernel function is defined as

$$K(x_i, x_j) = \tanh(\kappa x_i^T x_j - \delta)$$

This kernel is not as efficient for classification as are the other three. Indeed, one of the fundamental requirements on a valid kernel is that it must satisfy Mercer's theorem, and that requires that the kernel be

positive definite. However, the sigmoid kernel is not necessarily positive definite, and the parameters κ and δ must be properly chosen. In cases where the kernel is not positive definite, the results may be drastically wrong, so much so that the SVM performs worse than chance.

A noteworthy point is that for a certain range of values of κ and δ, the kernel behaves as a linear kernel, while for a certain other range of values of the same parameters, it takes the form of a RBF kernel.

V. Proposed Model

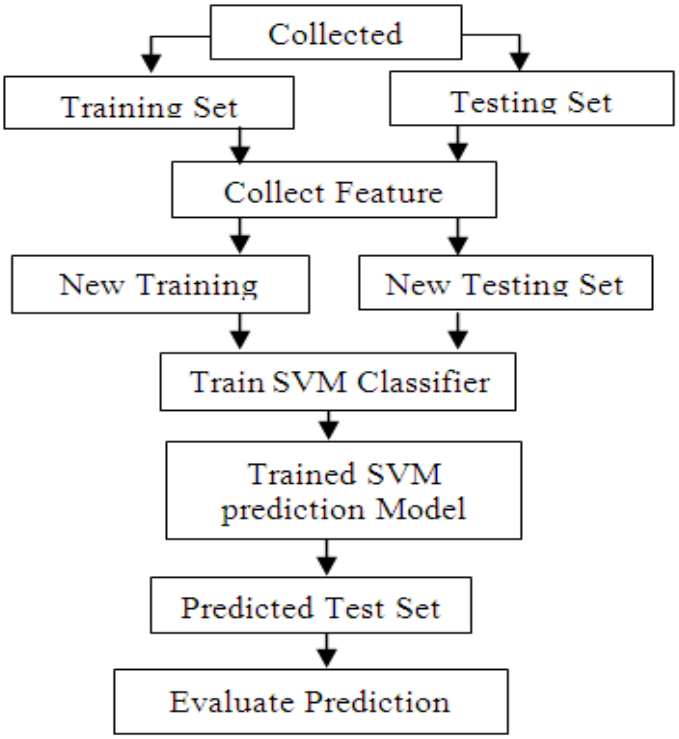

Fig. 4. System architecture of SVM based rainfallpredictive model

VI. Experimental Setup

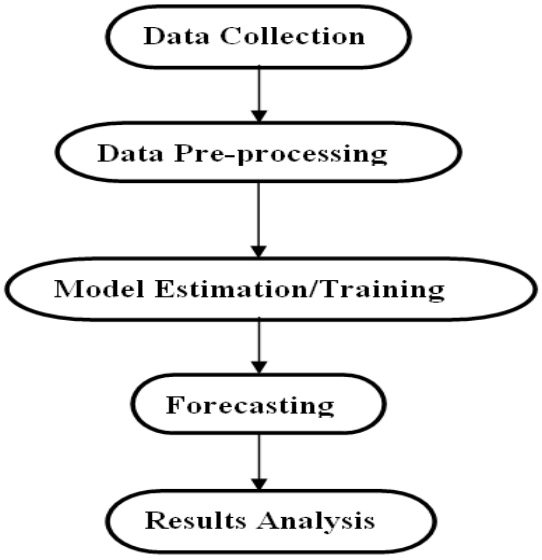

Fig. 5. Step of experimental setup

Figure 5 gives detail idea about the experimental setup and the steps followed in experiments. Figure 10 shows the flowchart representation of the overall methodology followed in the experiments. Details of each step and the experimental setup are discussed below.

6.1 Data Collection & Preprocessing

A monthly rainfall data in the period of 1993-2008 for Khurda District of Orissa were collected and some data preprocessing steps on raw set of monthly rainfall data as shown below:

1) Firstly, a monthly rainfall data were cleaned by filling in missing values with 0 values.
2) Secondly, a monthly rainfall data were normalized by min-max normalization into a specified range 0.0 to 1.0.

Out of all the data three number of individual data sets are created which can be used for Training, Validation & Testing. Each file must be in Text format. For Training 50 % of the total data set is used while remaining 25% of the data set is used for validation. Out of this 25% data set 12.5% data set should exchange with Training data set. So, that it is the combination of both known and unknown data set. The other remaining 25% data set which is known as unknown data set is used as Testing data set. For each data set first column must represent the output while other column represent the inputs which must be followed by the serial number of the input and a colon mark. Fixed no. of space or tab can be used between the columns in order to distinguish between them.

6.2 Training and Testing of SVM models

SVMdark tool was used for this part of the experiment. It uses SVMlight library for the SVM computations. This library implements optimization algorithms described in. It supports linear, polynomial, RBF and Sigmoid kernels. All of these kernels were tried in the experiments.

The decision on selecting appropriate values of d is basically by trial and error. However, based on a study by Ali and Smith (2003) using various sample sets with different umber of attributes and sizes, their experiments showed that the search space for d values should be ranged from 2 to 5. Therefore, in this project, the values of polynomial degree will be in the range of 0 to 5. Range of values used for these parameters is shown in table 1.

Table 1: Kernels parameter range

Parameters	**Start Range**	**End Range**
C	0	1000000
d (Polynomial kernel parameter)	0	5
s (Polynomial kernel parameter)	0	5
c (Polynomial kernel parameter)	0	5
gamma (Radial Basis kernel parameter)	-5	5
s (Sigmoid kernel parameter)	-5	5
c (Sigmoid kernel parameter)	-5	5

Initially through SVMdark we have to perform optimization, where we have to specify range of parameter values & suitable kernel. The output of optimization is produced in form of an excel file. Then, the range of these parameters will be narrowed as per the above table 4.5. Now the optimization progress goes on iteratively until the best parameters for a particular kernel types have been chosen which known as best SVM predictive model for current problem domain. Finally we will get the best values for these parameters for a particular kernel which will be used in the SVM model. Here optimization was applied for 10, 20, 30, 40 runs respectively. For every specified run on training and validation sets, MSE value was examined. The minimum MSE for each kernel run is selected which known as the best model.

Here we have selected each individual kernel functions with their best model one after the other, varying one parameter at a particular time while other parameters remains fixed in order to find the minimum MSE for every specified run. Each time using "optimize" file, the best suitable parameter values for corresponding kernel was selected. Then by taking these values, SVM model will be trained. Then trained model with their best parameter value will be used for prediction of unknown Test. The output of prediction is made in the prediction file which is in text format.

VII. Results

The output produced by each kernel at the time of training for each individual run along with their change in parameters are given below table 2:

Table 2: Comparison among all kernels based On MSE

Kernel Type	**10 runs**	**20 runs**	**30 runs**	**40 runs**
Linear	0.056654	0.050657	0.051041	0.050486
Poly	0.054938	0.050401	0.050494	0.058927
RBF	0.067852	0.057206	0.051019	0.051874
Sigmoid	0.067852	0.067852	0.067852	0.067852

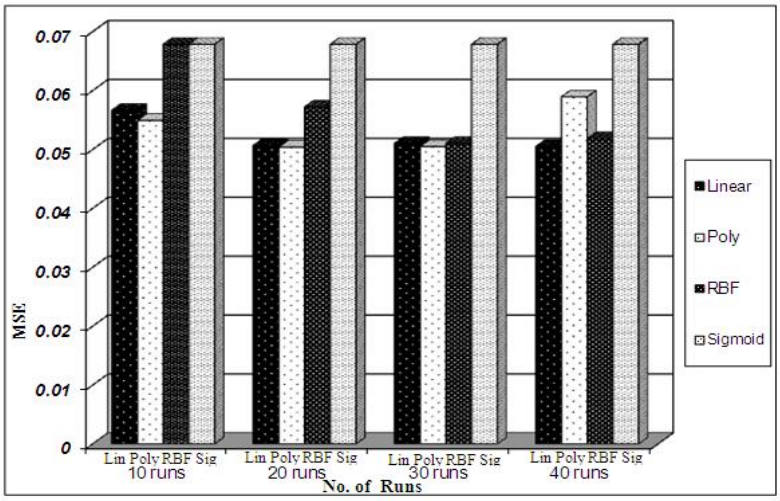

Fig. 6. Comparison of all kernels based on MSE

By observing the MSE result of each kernel of table 2 best run for each kernel is decided i.e. Linear kernel's 40 run, Polynomial kernel's 20 run, RBF kernel's 30 run & Sigmoid kernel's 10 run was selected for parameter change. Now the parameter change can be applied over each kernels best run (except linear) in order to get the lower MSE with their optimized parameter value.

After getting the optimized parameter value for each kernels best run, now the predictions can be made for each individual kernel by giving the optimize value of each parameter.

Now, average error percentage is calculated for each individual kernel prediction result by using equation (3):

Mod (Actual(Un-Normalized) - Predictions (De-Normalized), (Actual(Un-Normalized))) (3)

The average error percentage in each kernel while producing their result is :

Table 3: Comparison of Results

Kernels →	Linear 40 Runs	Polynomial 20 Runs	RBF 30 Runs	Sigmoid 10 Runs
Average Error %	15.04%	18.81%	16.15%	18.32%

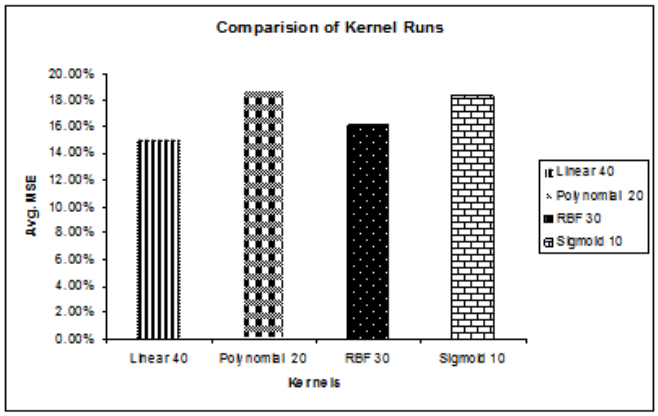

Fig. 7. Comparison of kernels with no. of runs

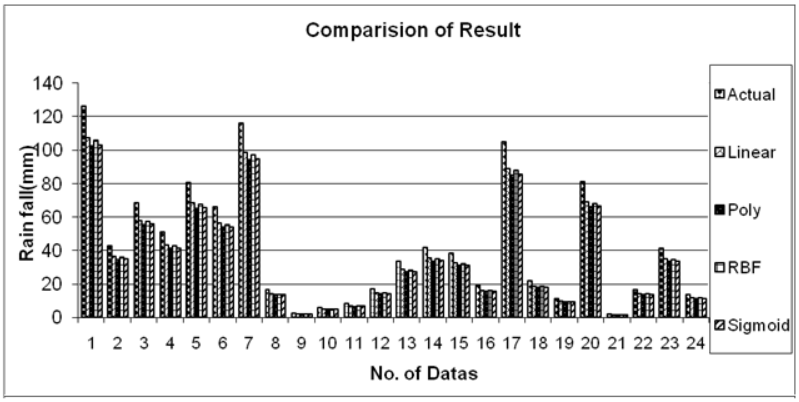

Fig. 8. Comparision of rainfall among different kernels

VIII. Conclusion

In this study, it has been demonstrated that the proposed approach Linear kernel could yield significantly higher prediction accuracy in comparison to other kernels of SVM by minimizing the average mean square error. There is no fixed rule in the choice of kernel function and its runs. But it is seen that the result produced at the time of training may not be the accurate i.e. optimize because after changing the parameter value for each kernel the other kernel (except the best kernel and run in training) may provide better result. Here though polynomial kernel function works generally well with non-separable data sets at the time of training by producing low MSE. But by increasing the degree of the function parameter, one can get zero mislead information. Since it is a machine learning technique so, the problem of underfitting & overfitting may usually arise which is very difficult to control. The other powerful kernel function is Gaussian kernel function. By controlling the value of ρ, zero misclassification can be attained as ρ controls the spread of the function.

Currently, work is moving in the direction of designing kernel functions and their parameter to help in control and reduce the attributes of a data set. This concept can be successfully useful for various data sets. The data set used for training, validation & testing should made in a general approach without any constraints.

References

[1]. http://en.wikipedia.org/wiki/Artificial_neural_network the details of Artificial Neural Network.
[2]. http://disi.unitn.it/~segata/FaLKM-lib/papers/ FaLKNR_ICCBR09_slides.pdf
[3]. Nello Cristianini and John Shawe-Taylor. An Introduction to Support Vector Machines and Other Kernel-based Learning MethodsCambridge University Press, March 2000.
[4]. Vautard, R., Yiou, P., and Ghil, M., 1992: Singular-spectrum analysis: A toolkit for short, noisy chaotic signals, Physica D, 58, 95-126.
[5]. Lisi, F., Nicolis, O., Sandri, M., 1995. Combining Singular-spectrum analysis and neural networks for time series forecasting. Neural Processing Letter 2 (4), 6-10.
[6]. N. E. Graham, J. Michaelsen, and T. P. Barnett, An investigation of the El Nino-Southern Oscillation cycle with statistical models - 1. Predictor field characteristics, J. Geophys. Res. 92, 1425 (1987); N. E. Graham, J. Michaelsen, and T. P. Barnett, An investigation of the El Nino-Southern Oscillation cycle with statistical models - 2. Model results, J. Geophys. Res. 92, 1427
[7]. Jayawardena, A. W. & Lai, F. 1994 Analysis and prediction of chaos in rainfall and stream flow time series. J. Hydrol. 153, 23-52.
[8]. Sharifi, M. B., Georgakakos, K. P., and Rodriguez-Iturbe, I.: Evidence of deterministic chaos in the pulse of storm rainfall, J. Atmos. Sci., 47, 888-893, 1990.
[9]. Sivakumar, B., Liong, S-Y., and Liaw, C. -Y.: Evidence of chaotic behavior in Singapore rainfall, J. Am. Wat. Re., 34, 301-310, 1998.
[10]. Islam, Liu, Q., S., Rodriguez-Iturbe, I., and Le., Y.: Phase-space analysis of daily streamflow: characterization and prediction, Adv. Water R., 21, 463-475, 1998.
[11]. Mukherjee, S., Osuna, E. & Girosi, F. 1997 Nonlinear prediction of chaotic time series using support vector machines. Neural Networks for Signal Processing-Proceedings of the IEEE Workshop. IEEE, Piscataway, NJ, pp. 511-520.
[12]. Dibike, Y. B. 2000 Machine learning paradigms for rainfall-runoff modelling. Hydroinformatics 2000, Iowa Institute of Hydraulic Research, Iowa, USA (CD-ROM).
[13]. Babovic, V., Keijzer, M. & Bundzel, M. 2000 From global to local modelling: a case study in error correction of deterministic models. Hydroinformatics'2000, Iowa Institute of Hydraulic Research, Iowa, USA (CD-ROM).
[14]. Liong, S. Y. & Sivapragasam, C. 2000 Flood stage forecasting with SVM. Submitted for publication in J. Am. Water Res. Assoc.
[15]. Vladimir N. Vapnik. The nature of statistical learning teory. Springer-Verlag New York.

Jyoti Ranjan Mohanty "Rainfall Prediction Using Support Vector Machine (SVM)." IOSR Journal of Computer Engineering (IOSR-JCE) 20.3 (2018): 06-13.