



**MARMARA UNIVERSITY**  
**INSTITUTE FOR GRADUATE STUDIES**  
**IN PURE AND APPLIED SCIENCES**



# **Implementing Support Vector Regression**

---

---

**method in electricity price forecasting**

**ALIASGHAR KAVIAN**

**MASTER THESIS**

Department of Computer Engineering

**Thesis Supervisor**

Assoc. Prof. Dr. Melih KIRLIDOĞ

**Thesis CO- Supervisor**

Prof. Dr. Seniye Ümit OKTAY FIRAT

**ISTANBUL, 2014**

---

---



**MARMARA UNIVERSITY  
INSTITUTE FOR GRADUATE STUDIES  
IN PURE AND APPLIED SCIENCES**



## **Implementing Support Vector Regression method in electricity price forecasting**

---

---

**ALIASGHAR KAVIAN**

**(524110005)**

**MASTER THESIS**

**Department of Computer Engineering**

**Thesis Supervisor**

**Assoc. Prof. Dr. Melih KIRLIDOĞ**

**Thesis CO- Supervisor**

**Prof. Dr. Seniye Ümit OKTAY FIRAT**

**ISTANBUL, 2014**

---

---

**MARMARA UNIVERSITY**  
**INSTITUTE FOR GRADUATE STUDIES**  
**IN PURE AND APPLIED SCIENCES**

ALIASGHAR KAVIAN, a Master of Science student of Marmara University Institute for Graduate Studies in Pure and Applied Sciences, defended his thesis entitled “Implementing Support Vector Regression method in electricity price forecasting”, on .2.2.....2.5..., 2014 and has been found to be satisfactory by the jury members.

**Jury Members**

Assoc.Prof. Melih Kırılıdoğ (Advisor)

Marmara University ..... (SIGN).....

Assoc.Prof. Bahar Sennaroğlu (Jury Member)

Marmara Üniversitesi ..... (SIGN).....

Assist.Prof. Fatma CORUT ERGİN (Jury Member)

Marmara Üniversitesi ..... (SIGN).....

**APPROVAL**

Marmara University Institute for Graduate Studies in Pure and Applied Sciences Executive Committee approves that ALIASGHAR KAVIAN be granted the degree of Master of Science in Department of Computer Engineering, Computer Engineering Program on ....2.....2.5... , 2014. (Resolution no: .2.2.4.7.44.2.2.).

**Director of the Institute**  
**Prof. Dr. Abdülkerim KAR**  
Müdür



## **ACKNOWLEDGMENT**

I would like to express my gratitude to several people. I would like to express my gratitude to my supervisors Melih Kırılıdoğ and Seniye Ümit OKTAY FIRAT for the useful comments, remarks, useful suggestions and engagement through the learning process of this master thesis; Also for the guidance is helping surpass all the trials that the researchers encountered. A special thanks to my family. Words cannot express how grateful I am to my mother and father for all of the sacrifices that they have made on my behalf and their endless love and support. Your prayer for me was what sustained me thus far. Furthermore I would like to thank Kenan Bayaz for introducing me to the topic as well for providing data.

# TABLE OF CONTENTS

ACKNOWLEDGMENT.....	i
TABLE OF CONTENTS.....	ii
ABSTRACT .....	v
ÖZET.....	vi
ABBREVIATIONS.....	vii
LIST OF FIGURES.....	x
LIST OF TABLES.....	xi
Chapter 1: INTRODUCTION .....	1
Chapter 2: GENERAL BACKGROUND AND LITERATURE REVIEW.....	9
2.1.Particle Swarm Optimization (PSO).....	9
2.1.1.Swarm intelligence.....	9
2.1.2.The social metaphor.....	9
2.1.3.The basic PSO algorithm.....	10
2.1.4.The continuous PSO.....	11
2.1.5.The binary PSO.....	12
2.1.6.Basic PSO algorithm pseudo code.....	12
2.1.7.Neighborhood topologies.....	13
2.1.8.Constriction factors.....	14
2.1.9.Parameter selection for the PSO.....	15
2.2.Related Studies About PSO.....	16
2.2.1.Parameter selection based on PSO.....	16
2.2.2.Feature selection based on PSO.....	26
2.2.3.Simultaneous parameter and feature selection based on PSO.....	34
2.3.Artificial Bee Colony.....	45
2.3.1.Swarm intelligence and behaviour of real bees.....	45
2.3.2.artificial bee colony (ABC) algorithm.....	49
2.3.3.Related works with ABC.....	51

2.3.4.Feature selection based on ABC.....	53
2.3.5.SVM parameter optimization based on ABC.....	56
2.4.v-SVR.....	60
<b>Chapter 3: MATERIAL AND METHOD.....</b>	<b>62</b>
3.1.Data Extraction.....	62
3.1.1.Data.....	62
3.1.2.Correlation.....	63
3.2.Fitness Definition.....	66
3.3.Simultaneous Parameter Determination And Feature Selection Based On PSO.....	67
3.4.IPSO.....	71
3.5.MPSO .....	71
3.6.The ABC Method For Simultaneous Parameter Determination And Feature Selection.....	72
3.6.1.The employed bee phase.....	72
3.6.2.The onlooker bee phase.....	73
3.6.3.Scout bees phase .....	73
<b>Chapter 4: DISCUSSION AND EVALUATION.....</b>	<b>75</b>
4.1.Experiments Of The Standard Version Of The PSO.....	75
4.2.Experiments Of The IPSO.....	80
4.3.Experiments Of The MPSO.....	83
4.4.Comparison Of The PSO Based Methods.....	84
4.5.The ABC Experiments.....	85
4.5.1.Test of limit value.....	85
4.5.2.Test of cross validation and dataset size.....	87
4.5.3.Testing parameters of ABC and comparison.....	92
4.6.Comparison Of ABC And PSO.....	99
<b>Chapter 5: CONCLUSION.....</b>	<b>102</b>
<b>REFERENCES.....</b>	<b>104</b>

**RESUME.....110**

## **ABSTRACT**

In deregulated electricity markets, electricity price prediction is a critical issue to develop risk management strategies and decision making process for market contributors. In this thesis, Support Vector Regression (SVR) used to train and predict electricity prices because of its outstanding performance in solve many real world problems. But in order to achieve best desirable results, SVR hyper-parameters should be determined appropriately. On the other hand, there are many features used in price forecasting and with high possibility some of them are irrelevant or redundant. Because of this reason it seems necessary to select relevant subset of features. The aim of this study is to predict day-ahead electricity price in Turkey electricity market with high accuracy and also with simple possible model. Feature subset selection as well as convenient parameter selection of the SVR can improve the SVR prediction accuracy considerably. Due to ability of the nature inspired heuristic (NIH) in both parameter determination and feature subset selection, this study attempt to use NIH techniques to SVR hyper-parameters determination and feature subset selection simultaneously. Artificial bee colony algorithm (ABC), used to simultaneous parameter determination and feature subset selection due to Turkey Day-Ahead electricity market price prediction with SVR. In order to figure out performance and efficiency of model, Particle Swarm Optimization algorithm (PSO) used for comparison. Results show that the ABC outperforms the PSO and gives better results.

## ÖZET

Düzensiz elektrik piyasalarında, risk yönetim stratejilerini geliştirmek ve pazar katılımcılar için karar verme süreci, elektrik fiyat tahmininde kritik bir konudur. Bu tezde, elektrik fiyatları eğitmek ve tahmin etmek için Destek Vektor Regresyon (SVR) yöntemi kullanılmıştır çünkü birçok gerçek dünya sorunlarını olağanüstü performansıyla çözmüştür. Ancak, en iyi istenen sonuçları elde etmek için SVR hiper parametreleri uygun şekilde belirlenmelidir. Diğer taraftan, fiyat tahmininde kullanılan birçok özellik vardır ve yüksek olasılıkla bazıları alakasız ya da gereksizdir. Bu nedenle, bu özellikler ilgili alt kümesini seçmek için gerekli görünmektedir. Bu çalışmanın amacı, gün öncesi Türkiye elektrik piyasasında elektrik fiyatını basit olası bir model ve aynı zamanda yüksek hassasiyetle tahmin etmektir. Alt kümesi seçimi özelliği gibi SVR'ın uygun parametre seçiminde de önemli SVR öngörü doğruluğunu artırabilir. Doğa esinli algoritmaların (NIH) parametre belirlenmesi ve alt kümesi seçimi özelliği gücü nedeniyle, bu çalışmada SVR hiper parametreleri belirlenmesi ve aynı anda alt kümesi seçimi özelliğinde NIH teknikleri kullanılmıştır. Yapay arı koloni algoritması (ABC), SVR ile Türkiye Gün Öncesi elektrik piyasa fiyatı tahmini dolayısıyla eşzamanlı parametre belirlenmesi ve alt kümesi seçimi özelliğiyle kullanılmıştır. Performans ve modelimizin etkinliğini anlama amacıyla, Parçacık Sürü Optimizasyonu algoritması (PSO) karşılaştırmak için kullanılmıştır. Sonuçlar ABC'nin PSO'dan daha iyi performans ve sonuçlar verdiğini göstermektedir.

## **ABBREVIATIONS**

**ABC** : artificial bee colony

**ABC-LSSVM** : artificial bee colony-least squares support vector machine

**ACO** : ant colony optimization

**ANN** : artificial neural network

**ARIMA** : autoregressive integrated moving average

**B-BPSO** : boolean binay particle swarm optimization

**BN** : number of employed bees

**BP** : back propagation

**BPNN** : back propagation neural network

**BPSO**: binary particle swarm optimization

**BPSO-SVM** : binay particle swarm optimization-support vector machine

**D(d)** : type of the day

**DBPSO** : discrete binary particle swarm optimization

**DBPSOKNN** : discrete binary particle swarm optimization K-nearest neighbor

**DBPSO-SVM** : discrete binary particle swarm optimization-support vector machine

**DE** : differential evolution

**EA** : evolutionary algorithm

**EMA** : exponential moving average

**eABC-LSSVM** : enhanced artificial bee colony-least squares support vector machine

**EC** : Evolutionary computating

**ECG** : electrocardiogram

**ES** : evolutionary strategy

**GA** : genetic algorithm

**GALS-SVM** : genetic algorithm least squares support vector machine

**GARCH** : generalized autoregressive conditional heteroskedasticity

**GA-SVM** : genetic algorithm-support vector machine

**GLCM** : gray level co-occurrence matrix

**GP** : genetic programming

**GRNN** : generalized regression neural network

**H** : humidity index

**HCC** : Hepatocellular carcinoma  
**HGA-SVR** : hybrid model of GA and SVR  
**HPSO** : hybridized the particle swarm optimization  
**HPSO-SVR** : hybridized the particle swarm optimization and support vector regression  
**IBPSO** : improved binary particle swarm optimization  
**IPSO** : improved particle swarm optimization  
**K-NN** : K-nearest neighbor  
**LP** : lagged price  
**LS-SVM** : least squares support vector machine  
**MA** : moving average  
**MAE** : mean absolute error  
**MAPE** : mean absolute percentage error  
**MBPSO** : modified binary version of the PSO  
**MCN** : maximum cycle number  
**MCP** : marginal clearing price  
**MPSO** : modified particle swarm optimization  
**MRI** : magnetic resonance imaging  
**MSE** : mean square error  
**NIH** : nature inspired heuristic  
**NMSE** : normalized mean squared error  
**PCA** : principal component analysis  
**PR** : pearson relative coefficient  
**PS-EA** : particle swarm inspired evolutionary algorithm  
**PSO** : particle swarm optimization  
**PSOLS-SVM** : particle swarm optimization least squares support vector machine  
**PSO-SVM** : particle swarm optimization-support vector machine  
**r** : degree of correlation  
**RBF** : radial base function  
**RBFNN** : radial basis function neural network  
**Rf** : rain fall  
**RMSE** : root mean square error  
**SBS** : sequential backward selection

**SFS** : sequential forward selection  
**SI** : Swarm intelligence  
**SLT** : statistical learning theory  
**SMA** : simple moving average  
**SN** : number of food sources  
**STLF** : short-term load forecasting  
**SVM** : support vector machines  
**SVR** : support vector regression  
**T<sub>a</sub>** : average temperature  
**T<sub>h</sub>** : highest temperature  
**T<sub>l</sub>** : lowest temperature  
**TMP** : trans membrane potential  
**UCI** : university of California.Irvine  
**VC dimension** : Vapnik–Chervonenkis dimension  
**W(D)** : type of the week  
**WMA** : weighted moving average  
**W<sub>p</sub>** : wind power  
**v-SVM** : Nu support vector machine  
**v-SVR** : Nu support vector regression  
**ρ** : degree of correlation

## LIST OF FIGURES

Figure 2.1 Structure of the proposed method in [12].....	17
Figure 2.2 Flow Chart of the proposed method in [12].....	18
Figure 2.3 Flowchart of proposed method in [23].....	20
Figure 2.4 Flowchart of proposed method in [57].....	22
Figure 2.5 Flowchart of proposed model in [59].....	24
Figure 2.6 Flowchart of proposed model in [59].....	25
Figure 2.7 Representation of the boolean opration to reset global best based on [15]....	28
Figure 2.8 Flowchart of the model built in [60].....	32
Figure 2.9 Flowchart of the model built in [63].....	33
Figure 2.10 Representation of a prticle in [42].....	36
Figure 2.11 Representation of a prticle in [17].....	37
Figure 2.12 Architecture of the model proposed in [17].....	38
Figure 2.13 Representation of a prticle in [39].....	39
Figure 2.14 General process of proposed model by [39].....	40
Figure 2.15 Representation of a prticle in [19].....	41
Figure 2.16 Representation of a prticle in [18].....	43
Figure 2.17 Process of the HPSO-SVR model according to [18].....	44
Figure 2.18 A graphical sample for behavior of bees based on [80].....	48
Figure 2.19 Block diagram of ABC based feature selection in [88].....	55
Figure 2.20 process of of eABC-LSSVM in [89].....	57
Figure 2.21 Flow of proposed method in [92].....	59
Figure 3.1 Representation of a prticle in proposed PSO.....	68
Figure 3.2 Flowchart of computing fitness.....	69
Figure 3.3 shows the procedure of the PSO.....	70
Figure 3.4 Representation of food source in ABC algorithm.....	73
Figure 4.1 trapping in a local optimal.....	80
Figure 4.2 trapping in a local optimal.....	82
Figure 4.3 effect of population size on results.....	94
Figure 4.4 comparison of one and two month data.....	96
Figure 4.5 Comparison of ABC and PSO.....	101

## LIST OF TABLES

Table 2.1 Represent three different sets of parameters proposed before.....	16
Table 2.2 Datasets and their characteristics used to evaluated model in [42].....	37
Table 2.3 Characteristics of used datasets according to [19].....	42
Table 3.1 Spearman’s correlation coefficient results for features.....	65
Table 4.1 Results of PSO with parameters based on [47].....	76
Table 4.2 Results of PSO with parameters based on [41].....	77
Table 4.3 Results of PSO with parameters based on [44].....	78
Table 4.4 Actual MSEs of 10 runs of PSO.....	79
Table 4.5 Results of IPSO.....	81
Table 4.6 Results of MPSO.....	83
Table 4.7 Comparison of proposed PSO based methods.....	85
Table 4.8 Analyzing ABC with different limit values.....	87
Table 4.9 Analyzing cross validation and separate dates effect with 30 population size and 5 fold cross validation.....	89
Table 4.10 Analyzing cross validation and separate dates effect with 40 population size and 5 fold cross validation.....	90
Table 4.11 Analyzing cross validation and separate dates effect with 50 population size and 5 fold cross validation.....	90
Table 4.12 Analyzing cross-validation and separate dates effect with 30 population size and 10 fold cross validation.....	91
Table 4.13 Analyzing cross validation and separate dates effect with 40 population size and 10 fold cross validation.....	91
Table 4.14 Analyzing cross validation and separate dates effect with 50 population size and 10 fold cross validation.....	92
Table 4.15 Population size 10.....	97
Table 4.16 Population size 20.....	97
Table 4.17 Population size 30.....	97
Table 4.18 Population size 40.....	98
Table 4.19 Population size 50.....	98
Table 4.20 Population size 60.....	98

Table 4.21 Population size 80.....	99
Table 4.22 Population size 100.....	99
Table 4.23 Comparison of ABC with PSO based Mehtods.....	100

## Chapter 1: INTRODUCTION

In today's auction-based electricity markets, efficient generation, consumption of electricity and reduction in energy prices are three aims of power industries in deregulated electricity market. The consumers' goal is to schedule their energy consumption ahead of time and minimize their expenses [2]. Energy suppliers aim in selling all generated energy and maximize their profit [1].

Interplay between suppliers and consumers take place in the pool where power supplier offer their generation amount and accordingly price bids as consumers submit their bids too. After the day-ahead market bid period closes, the system operator calculates the day-ahead schedule and the day-ahead MCPs (marginal clearing prices) based on the bids, offers, and schedules submitted based on least-cost, security constrained unit commitment [3], then forward for every hour of the next work day accordingly. Market contributors expect to take advantage of price forecasting to develop optimal bidding strategies. Thus, day-ahead electricity price prediction importance for market contributors in deregulated electricity market is as below:

- i. It helps market participants' decision making.
- ii. It is a critical issue aids to develop risk management strategies.
- iii. It helps to strategy development process for market contributors on how to increase their profit.

Electricity price prediction is a critical issue to risk management strategies, decision making and strategy development process for market contributors in deregulated electricity market. Electricity price have some particular characteristics as below:

- i. non-linear
- ii. time variant
- iii. non-stationary
- iv. volatile signal with multiple periodicity
- v. high frequency components significant outliers

Also there are some complications in dealing with electricity market price. It is impossible to store electricity. In electricity market, there are hourly, daily and seasonal ambiguities. Seasonal uncertainty could be rapid changes in weather, fuel price and

behavior of market contributors. The mentioned reasons make electricity prices much more difficult.

There are many techniques for day-ahead electricity price forecasting. Most of them are based on statistics and/or artificial intelligence. In the following some of studies will be discussed: The autoregressive integrated moving average (ARIMA) analysis method was used to forecasting next day electricity prices in [4], a model based on ARIMA and wavelets was proposed by [5]. Garcia et al.[6] proposed generalized autoregressive conditional heteroskedasticity (GARCH) model due to forecast price in the California and the Spanish electricity market, as well as good results was achieved. As regards for volatile and heteroskedastic time series, prediction results are not satisfactory. Thus, they fit linear relationships and mentioned methods are not suitable for disposing of non-linear relationship. In other words, they show good performance in a stable market but not for modeling more complex forecasting like electricity market price.

One of the methods based on artificial intelligence that acts well to predict non-linear problems is Artificial Neural Network (ANN). Other reasons that cause receives more attention are its clear model, easy implementation, good flexibility and robustness. Thus ANN is appropriate than time series for complex price forecasting. ANN can predict nonlinear functions accurately where inputs include historical prices and the influential information. This is an advantage because in mentioned methods before, they mostly use historical prices as features and did not use the influential information. After training and validating through a given data, the ANN obtains output by putting new data into trained model. The ANN was applied in many studies recently [7-9]. The basic rule of training ANN is to minimization of training error, but there is no considering the complexity of the model. In this condition, by increasing complexity of the model, the degree of freedom decreases. Mentioned issue can cause overfitting or underfitting dilemma and also choosing appropriate parameters of the ANN is a great challenge too.

In recent years, Support Vector Machines (SVM) attracted more attention for their outstanding performance. SVM inspired by VC dimensional theory and statistical learning theory (SLT) [20] and also it can solve both classification and regression problems. In comparison with other machine learning techniques such as ANN, SVM has some considerable advantages as below:

- i. SVM solves a quadratic programming problem and make sure when algorithm provides an optimal solution, it is unique.
- ii. SVM derives a sparse and robust solution by maximizing the margin between the two classes [22].
- iii. SVM is based on the structural risk minimization principle.
- iv. Different kernel functions can be use due to solve linear and nonlinear problems.
- v. The kernel function can be useful to prevail the “curse of dimension”.

In the training phase of SVM there are two goals: minimization of training error and complexity of the model. Due to solve regression problems such as electricity market price prediction, Support Vector Regression (SVR) can be suitable based on mentioned advantages over other approaches mentioned before. SVR is a version of the SVM used for solving regression problems. In this study, SVR preferred because of mentioned characteristics of data and advantages over time series and ANN models.

SVR has been applied to many fields such as optimal control, image segmentation and etc. In SVR, there are a set of hyper-parameters must be chosen such as kernel parameter  $\gamma$ , the regularization constant  $C$  and epsilon  $\epsilon$ . Selection of appropriate hyper parameters for SVR has great impact on performance. Schölkopf et al in [21] proposed selecting  $\nu \in (0, 1)$  instead of parameter  $\epsilon$  that is called  $\nu$ -SVR. The new parameter ( $\nu$ ) can control the number of support vectors and training errors efficiently. The theoretical interpretation is that  $\nu$  is an upper bound on the fraction of margin errors, although  $\nu$ -SVR produces a more generalized regression than other machine learning methods such as artificial neural networks [22].

In dealing with SVM and/or SVR, there are two issues that influence the performance and accuracy of the model considerably:

- i. Hyper parameters determination
- ii. Feature subset selection.

Selection of appropriate hyper parameters for SVR has great impact on performance of the model. Early studies figure out these parameters based on trial and error procedure and also grid search. But after those, evolutionary computation techniques such as swarm intelligence and genetic algorithms applied to parameter selection in many

studies due to achieve best possible performance [10-13]. In addition, evolutionary computing techniques such as particle swarm optimization (PSO) perform better than other methods such as traditional parameter selection and grid search [12, 23]. For example in [12] and [23], the PSO used to parameter determination of SVR and compared with traditional SVR whose obtained parameters with trial and error and grid search, it provided higher prediction precision and spends even less time on parameters selection. Wang et al. [24] has proposed a model based on support vector regression with differential evolution algorithm to deal with the problem of annual load forecasting. Jiang et al.[25] used SVR in Solving the Inverse ECG Problem. It used three different optimization methods: genetic algorithm (GA), differential evolution (DE) algorithm, and particle swarm optimization (PSO) to parameter determination of the SVR and compared results. The experimental results show that these three optimization methods are well performed in finding the proper parameters of SVR and can yield good generalization performance in solving the inverse ECG problem and Moreover, compared with DE and GA, PSO algorithm is more efficient in parameters optimization and performs better in solving the inverse ECG problem, leading to a more accurate reconstruction of the trans membrane potentials (TMPs) [25]. Wu et al.[26] proposed a novel hybrid model of GA and SVR, HGA-SVR, for type of kernel function and kernel parameter value optimization in SVR in the case of forecasting maximum electrical daily load. Zhang et al.[27] used ant colony optimization (ACO) algorithm for parameter optimization of support vector machines.

Due to achieve better performance in regression models, feature subset selection is an important issue must be considered too. It is beneficial to restrict number of input features to produce a good predictive and less computationally intensive model [14]. In other words, SVR can be realized properly with appropriate and small number of feature subset. Potentially, many features could be used to predict electricity price and to achieving desirable prediction accuracy, feature subset selection seems inevitable. In the following feature selection methods, their importance and some relevant concepts will be discussed.

Feature selection is the process of choosing relevant features or attributes in construction of model. In data mining, machine learning and statistics, feature selection known as attribute selection, variable subset selection or variable selection. The aim of

feature selection techniques is to eliminate redundant features which supply no more beneficial information than selected features. And also returns a subset of the features. There three advantages of feature selection as below:

- i. Reduction of overfitting
- ii. Less training time
- iii. Improvement of model interpretability

Feature selection is a subset of field of dimension reduction [28]. For high-dimensional datasets (usually number of dimensions is more than 10), most of times dimension reduction applied formerly due to avoid the effects of curse of dimensionality. The curse of dimensionality refers to different effects of analyzing data in high-dimensional spaces. These effects do not happen in low-dimensional data. This problem addressed in domains such as numerical analysis, machine learning, data mining and databases. The volume of the space increase fast and data become sparse when dimensionality increases which is common issue of all mentioned domains. The sparsity make problem for methods need statistical significance. Because of provide confidential results based on statistics, with increasing number of dimensions, amount of data for supporting outcome grows exponentially. Also organizing and searching data often relies on detecting areas where objects form groups with similar properties; in high dimensional data however all objects appear to be sparse and dissimilar in many ways which prevents common data organization strategies from being efficient [29]. Based on intrinsic dimension concept, by adding redundant dimensions to low-dimensional data it could be turned into high-dimension data and also high-dimension data could be turned into low-dimension without remarkable loss of information. This can be done by dimension reduction techniques like principal component analysis (PCA). Furthermore there is direct relationship between dimension of data and number of samples which is with increasing dimensionality for a fixed number of samples, the predictive ability reduce and this concept known as the Hughes effect [30] or Hughes phenomenon.

Both supervised and unsupervised feature selection techniques can classify into three major categories:

- i. Filter approaches: they attempt to eliminate irrelevant features before using in leaning process. In this way, after analyzing data, most relevant features choose

to training algorithm. During selection, feedback of the algorithm performance is not needed.

- ii. Wrapper methods: they use learning algorithm itself due to find appropriate features. In this way, different feature subsets are evaluated by prediction accuracy.
- iii. Embedded approaches: In these methods process of feature selection is part of learning algorithm itself.

In this part of study, feature selection method categories into two major classes: traditional feature selection and evolutionary computation approaches. In the following, some techniques and samples of each one of mentioned categories will be discussed.

Traditional feature selection methods:

- i. Relief is a filter method. It specifies a weight for each feature that defines its relevance to target. This method tries to find all relevant features but it does not consider about redundant features [31].
- ii. Sequential forward selection (SFS) is a wrapper method and it is based on greedy search. It starts with no features and adds features to the subset until adding features do not increase performance. In this way, a feature cannot be eliminated when it was selected. Mentioned problem is a disadvantage of this method.
- iii. Sequential backward selection (SBS) is a wrapper method and it is based on greedy search. It starts with all features and remove features from the subset until removing features do not increase performance. In this way, a feature cannot be added when it was eliminated before. This method like SFS has a same disadvantage too.

Evolutionary computation (EC) and nature inspired heuristic (NIH) for feature selection:

Rather than statistical methods to extract appropriate subset of features, EC and NIH techniques used to feature subset selection too. Most of studies tried to improve accuracy of classification and claimed to achieve better performance [15, 16]. EC cover vast area of studies such as genetic algorithms (GAs), genetic programming (GP), ant etc. In the following some of EC and NIH techniques used in feature selection problem will be mentioned briefly.

- i. Memetic Algorithms

- ii. GP
- iii. ACO
- iv. GA
- v. Simulated annealing
- vi. Differential Evolution (DE)
- vii. PSO
- viii. Artificial bee colony (ABC)

Sometimes due to limitations or acquire better results mentioned algorithm combined with other methods such as Zhu et al. [32] proposed a feature selection model based on memetic algorithm. The results show this method outperforms GA and other algorithms used in comparison and combination of ACO and rough set theory proposed in [34] for feature selection problem.

In addition, one highly relevant feature can be redundant when group with other features, therefore eliminating these features can reduce unnecessary complexity and on the other hand a weekly relevant feature can become highly relevant when working with others. In other words, solution depends on the combination of selected features which makes feature selection a NP-Hard problem. Thus, metaheuristic algorithms like PSO, ACO and evolutionary algorithms (EA) are suitable for this kind of problem because of randomized nature of them [36]. Metaheuristic algorithms can find appropriate solutions without test all possible combination or search space.

The PSO received more attention in the field of feature selection. Because particle swarm optimization is powerful, easy to implement, and computationally efficient [41]. But standard version of the PSO can converge to local optimal very fast (premature convergence) and when it happen probability of avoiding trapping in a local optimal is not high enough. Because of this problem many studies proposed some modifications due to avoiding to trapping in a local optimal. For example, Chuang et al. [37] proposed a technique for feature selection based on global best (gbest) which gbest will reset to zero after predefined number of iterations that gbest remain unchanged. Chuang et al. [38] proposed a modified version of the PSO for feature selection which reinitializing worst particle in the swarm when gbest has not improved for predefine number of iterations. There are several studies tried to overcome this problem of PSO and used

different techniques.

Feature subset selection as well as convenient parameter selection of the SVR can improve the SVR prediction accuracy considerably. Due to great performance of the Evolutionary Computing in both parameter determination and feature subset selection, in recent years some studies tried to feature subset selection and parameter optimization simultaneously. Most of them are based on genetic algorithm (GA) and the PSO [17, 18, 19, 39, 42]. Most of approaches are modified version of PSO and based on discrete PSO. Outcome of these methods compared with traditional SVR, GA, Other versions of PSO and other parameter determination or feature selection methods. Results show better accuracy or less error with simple model, low complexity and less experimental process time generally. Results of GA and PSO are quite close. Likewise both of them encounter difficulties in dealing with high dimension problems with too many local optimal.

The aim of this study is to build a simple and effective model for electricity price prediction base on v-SVR. Artificial Bee Colony (ABC) algorithm acts better in high dimension problems as well as its ability to avoid trapping in a local optimal are two reasons of choosing it for this study. In this study, the ABC algorithm used to simultaneous feature subset selection and parameter determination for training v-SVR in order to prediction of day-ahead electricity of turkey. After a brief description of v-SVR and PSO algorithm, the ABC algorithm will be illustrated. Then data used in experiments will be explained. Then proposed method will shows up. After all results of the ABC, PSO and comparison of them will be discussed.

## **Chapter 2: GENERAL BACKGROUND AND LITERATURE REVIEW**

### **2.1. Particle Swarm Optimization (PSO)**

#### **2.1.1. Swarm intelligence**

Swarm intelligence (SI) is the collective behavior of decentralized, self-systems, natural or artificial. The concept is employed in work on artificial intelligence [99]. Swarm intelligence systems consist of a crowd of simple individuals interacting locally with each other and also with their environment. It is usually inspired from nature. The individuals or agents follow simple rules and although there is no centralized control structure about behavior of agents, local and certain degree of random and Agents are self-organized. Interactions between such agents with mentioned characteristics lead to shaping "intelligent" global behavior that is unknown to the individual agents. In the following there are some examples of SI:

- i. Ant colony optimization
- ii. Artificial bee colony algorithm
- iii. Artificial immune systems
- iv. Bat algorithm
- v. Intelligent water drops
- vi. Particle swarm optimization

.There is no clear definition of swarm intelligence. In fact, it is called to a multi-agent system with self-organized behavior that shows some kind of intelligent behavior. Swarm intelligence refers to vast set of algorithms and used in many applications such as robotics and forecasting.

#### **2.1.2. The social metaphor**

The Particle Swarm Optimization algorithm or briefly PSO that described in [41], is a population based optimization algorithm and it is inspired by biology. In other words, It was inspired in the way of population of agents move to predefined objectives. Each agent or individual in the crowd attempt to move towards the best (fittest) known position to them and to their informants. Informants are the set of individuals that are individual's social circle. The main goal here is to minimize or maximize a fitness

function. Usually it is related to the evolutionary computation (EC) techniques, basically with genetic algorithms (GA) and evolutionary strategies (ES), but there are some differences with those techniques.

The PSO is a population-based algorithm. It means that a set of potential solutions give out to approach a suitable solution or solutions for a defined problem. The objective of the optimization method is finding the global optimum of a real-valued function (fitness function) defined in a search space.

Here is summary of the social metaphor for the PSO algorithm: each agent or individual that is part of population or society hold an opinion that it is part of a "belief space" (the search space) shared by every individual. Individuals can modify this "opinion state" based on three factors:

- i. Environmental knowledge (individual's fitness value)
- ii. Previous history of states for individual (individual's memory)
- iii. The previous history of states of the individual's neighborhood

There are several ways to define an individual's neighborhood. There are several topologies for neighborhood (full, ring, star, etc.). The individuals in the population try to adapt their opinion state to the ones that are more successful among their social network. After some time, a culture arises, in which the individuals hold opinions that are close together.

### **2.1.3. The basic PSO algorithm**

Each individual is called a "particle", and it can move in a multidimensional space that indicates the belief space. Particles have memory, so they can preserve part of their previous state. There is no constraint to sharing the same point in belief space for particle. But their individuality must be retained. Movements of particles are affected by an initial random velocity and two randomly weighted parameters. Initial random velocity and two randomly weighted parameters cause following two effects on the particles:

- i. Each particle has a tendency to return to best previous position in the population.
- ii. Each particle has a tendency to move towards the neighborhood's best previous position.

### 2.1.4. The continuous PSO

There are two versions of the basic Particle Swarm Optimization algorithm. The belief space in the "continuous" version is based on a real-valued multidimensional space. The "continuous" version of the PSO gives out the position of each and every particle using the equations (1) and (2):

$$v_{id}^{t+1} = w \cdot v_{id}^t + c_1 \cdot \psi_1 \cdot (p_{id}^t - x_{id}^t) + c_2 \cdot \psi_2 \cdot (p_{gd}^t - x_{id}^t) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \quad (2)$$

$v_{id}^t$ : Component in dimension d of the  $i^{\text{th}}$  particle velocity in iteration t.

$x_{id}^t$ : Component in dimension d of the  $i^{\text{th}}$  particle position in iteration t.

$c_1, c_2$ : Constant weight factors.

$p_i$ : Best position achieved so long by particle i.

$p_g$ : Best position found by the neighbors of particle i.

$\psi_1, \psi_2$ : Random factors in the [0,1] interval.

w: Inertia weight.

Value of  $p_g$  depend on selection of neighborhood type. In the basic PSO algorithm either gbest or local best (lbest) neighborhood is used. All the particles are investigated when using the global neighborhood for calculating  $p_g$ . But in the case local neighborhood, neighborhood is form by a predefined number particle among whole population and also the local neighborhood of specific particle does not change during the iteration of algorithm.

A constraint ( $v_{\max}$ ) is imposed on  $v_{id}^t$  to ensure convergence. Its value is usually kept within the interval  $[-x_{id}^{\max}, x_{id}^{\max}]$ , being  $x_{id}^{\max}$  the maximum value for the particle position [43]. Due to better global search large number for inertia weight (w) is suggested, and a small inertia weight favors local search. Sometimes inertia weigh decreased linearly along the iteration of the algorithm and also starting at initial value close to 1 [43, 44]. Clerc and Kennedy [45] suggest an alternative formulation of equation 1 and adds a constriction coefficient that replace the velocity constraint ( $v_{\max}$ ).

The Particle Swarm Optimization algorithm needed to tune of some parameters:

- The individual and sociality weights ( $c_1, c_2$ )
- The inertia factor ( $w$ )

References [41, 43, 44, 45, 46, and 47] contain theoretical and empirical studies to help in selection of appropriate values for mentioned PSO parameters.

### 2.1.5. The binary PSO

Binary version of PSO algorithm has been proposed in [41, 48]. In this version of the PSO algorithm, particle's position does not demonstrate with a real value but with either 0 or 1 binary value. As the probability distribution for the position, the logistic function of the particle velocity is used and position of the particle in a dimension is generated randomly using that distribution. Equation (3) is used to update the particle position:

$$x_{id}^{t+1} = \begin{cases} 1: \text{if } \psi_3 < \frac{1}{1 + e^{-v_{id}^{t+1}}} \\ 0: \text{otherwise} \end{cases} \quad (3)$$

$v_{id}^t$ : Component in dimension d of the  $i^{\text{th}}$  particle velocity in iteration t.

$x_{id}^{t+1}$ : Component in dimension d of the  $i^{\text{th}}$  particle position in iteration t+1.

$\psi_3$ : Random factor in the [0,1] interval.

In a binary version of the PSO there are no individual and social influences ( $c_1=c_2=0$ ). It performs a random search on the space. It means that in each dimension there is 50% chance of being a zero or one. Parameter selection for a binary PSO is not critical like the continuous PSO.

### 2.1.6. Basic PSO algorithm pseudo code

The pseudo code of the basic PSO can be written as follows with respect to [41]:

I) For each particle:

*Initialize particle*

II) Do:

a) For each particle:

- 1) Calculate fitness value
- 2) If the fitness value is better than the best fitness value (*pbest*) in history
- 3) Set current value as the new *pbest*

End

b) For each particle:

- 1) Find in the particle neighborhood, the particle with the best fitness
- 2) Calculate particle velocity according to the velocity equation (1)
- 3) Apply the velocity constriction
- 4) Update particle position according to the position equation (2)
- 5) Apply the position constriction

End

*While maximum iterations or minimum error criteria is not attained*

At the first step or in the initialization step, dimensions for each position start with random numbers and velocities can be initialized with random numbers or set to 0. In the original PSO algorithm, particles' velocities on each dimension are limited to maximum velocity value represented by  $V_{max}$ . For each dimension of a position,  $V_{max}$  can be specified by the user and when the sum of accelerations would cause the velocity on that dimension to exceed  $V_{max}$ , then the velocity on that dimension is limited to  $V_{max}$ . This is a mechanism to prevent the incident of "swarm explosion". *pbest* is abbreviation of previous best particle.

### **2.1.7. Neighborhood topologies**

In the original version of PSO, there are two different kinds of neighborhoods were specified:

- i. *gbest*
- ii. *lbest*

In the *gbest* swarm, all the particles in a population are neighbors for all other particles. Therefore, the position of the best particle in the swarm is used to update equation in term of velocity. In the *gbest* swarm convergence is fast because all particles are drawn to the best part of the search space altogether. In this situation, the PSO algorithm can be trapped in a local optimum, if found best particle is far from the global optimum solution.

In the lbest swarm, the velocity of the particle can be affected by a specific number of particles in the neighborhood. Thus, the convergence is slower than gbest and has greater chance to find global optimum.

In both of the mention methods, the relations among particles do not depend on their positions in the search space and external relationships.

### 2.1.8. Constriction factors

The aim of this part is to illustrate details and variation that have been studied on the basic PSO algorithm due to improving of some performance issues of the basic PSO. This part of study is based on [49] and [46].

Kennedy et al.[41] introduced the clamping effect to prevent the incident of "swarm explosion". The particle velocity can grow unbounded even if it is around an optimum when there is no restriction defined on the maximum velocity of the particles and also on each iteration, its distance to optimum increasing. In some studies, the velocity clamping parameter  $v_{max}$  was set to the maximum values for the specific position in the search space called  $x_{max}$ . Subsequent studies show that this system was not enough to properly control the particle's velocity. In comparison with evolutionary computing techniques in [50], the PSO algorithm could quickly find the area of optimum but had a problem of perform a fine grained search of that area. In the other words, it had trouble to adjust the velocity to lower values. In the further version of the PSO algorithm [45], equation (4) shows modified velocity equation that is used and the velocity clamping technique is not needed anymore.

$$v_{id}^{t+1} = x(w \cdot v_{id}^t + c_1 \cdot \psi_1 \cdot (p_{id}^t + x_{id}^t) + c_2 \cdot \psi_2 \cdot (p_{gd}^t - x_{id}^t)) \quad (4)$$

$w$  is called inertia weight

$x$  is the constriction factor

$c_1$  is the cognitive parameter

$c_2$  is the social parameter

Mixture of above parameters specifies the convergence properties of the PSO algorithm.

### 2.1.9. Parameter selection for the PSO

In the first version of the PSO algorithm there are three values that must be selected appropriately and they are  $c_1$ ,  $c_2$ , and  $v_{\max}$ . These values influence the convergence speed and ability to find optimum solutions. For different kinds of problems, it may be better to choose different values according to that problem. Many studies have been done to select appropriate combination of values for covering vast area of problems. Equations (5) and (6) are proposed for the constricted version of the PSO by [45] :

$$x = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4 \cdot \varphi}\right|} \quad (5)$$

$$\varphi = c_1 + c_2 > 0.4 \quad (6)$$

The inertia weight determines the influence of velocity of the particle on the velocity in the next iteration:

- For the  $w=0$  only  $p_i$  and  $p_g$  positions can influence the velocity of the particle. Low inertia weight favors exploitation or local search. It means that the particle may change its own velocity immediately if it is far from its best known position.
- For higher value of  $w$ , rate of changing velocity for particle is lower. In the other words, high inertia weights favor exploration or global search.

There are two studies about inertia weight in [43, 44] and they proposed a decaying inertia weight. At first due to better global search for the algorithm and later to better local search. They suggest selecting a value  $w \in [0.8, 1.2]$  for inertia weight in order when it is not reduced with time. The cognitive and social factors are not critical for algorithm but selection of better values cause a better performance of the algorithm. Selection of the cognitive and social factors, have impact on the algorithm in terms of speed of convergence and passing local minimum. In the case of selecting appropriate parameters for the PSO, there several studies that looks sufficient for usual benchmark functions. Table 2.1 demonstrate three of best found parameters combination for the PSO that used in many studies.

**Table 2.1** Represents three different sets of parameters proposed before

$c_1$	$c_2$	$w_{init}$	$w_{final}$	Reference
2.0	2.0	1.0	1.0	[41]
2.0	2.0	0.9	0.4	[44]
1.4962	1.4962	0.7968	0.7968	[47]

Theoretical studies about convergence were performed in [45] and [47]. In the later studies about parameter selection of the PSO, a heuristic procedure for parameter selection and tuning was proposed based on convergence speed of different combination of parameters. By combination of the constraint and the velocity clamping mechanisms, best results were achieved using  $v_{max}=x_{max}$  according to [51].

## 2.2. Related Studies About PSO

### 2.2.1. Parameter selection based on PSO

There are many studies about parameter optimization for SVMs because selecting appropriate hyper parameters of SVM has great impact on performance and accuracy of it. Parameter selection for Support vector machines is one of the critical issues that attracted by many studies. Different algorithms applied to solve parameter selection of Support vector machines such as Evolutionary Algorithms, Genetic algorithms, Swarm Intelligence and etc. to solve many real world problems that each one has its own advantages and disadvantages. Also there are many studies have been done in the case of the PSO algorithm for a vast area of problems. In this part of study, some of applications of the PSO algorithm integrated with Support Vector Machine have been illustrated.

Qun et al.[12] proposed a parameter selection method for Support Vector Regression based on a modified version of the PSO. In the original version of the PSO does not use particles efficiently over the search space. By analyzing velocity and position equations, it is clear that particles follow the global best, therefore probability of sticking in a local minimum is high and it is too hard to break away from that position. The mentioned phenomena described in [52]. To overcome this problem there are two solutions:

- i. Increasing number of particles
- ii. constraining particles from pursuing the best position

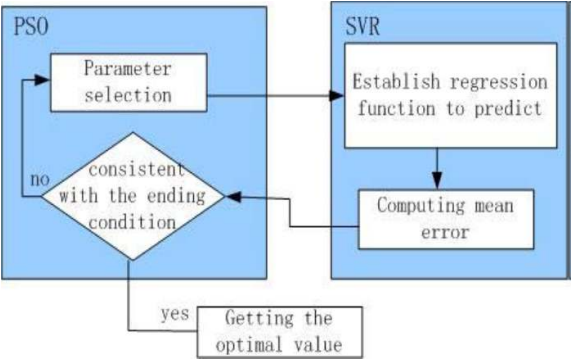
Based on mentioned reasons above, Qun et al.[12] proposed the following modified PSO briefly and exactly as follow:

Add n particles to the group, and the velocity and position is given randomly. This method does not affect the old particles pursuing the best point arrived by group to grantee the astringency of PSO but also ensure the particles searching in a larger area by increasing numbers of particles and setting the velocity and position of the new particles randomly [12].

Figure 2.1 that exactly copied from [12] represents the structure of proposed method and Figure 2.2 shows flow chart of the mentioned method. In this study, libsvm public source code for support vector machines [101] has been used with epsilon SVR and RBF kernel type. The SVR parameters that have been selected are:

- i. The trade off variable C
- ii. e
- iii. Kernel parameters s2

There is no maximum iteration size and it is measured by time (200 second). The results of tests shows better performance in searching time and also on the identification accuracy.



**Figure 2.1** Structure of the proposed method in [12]

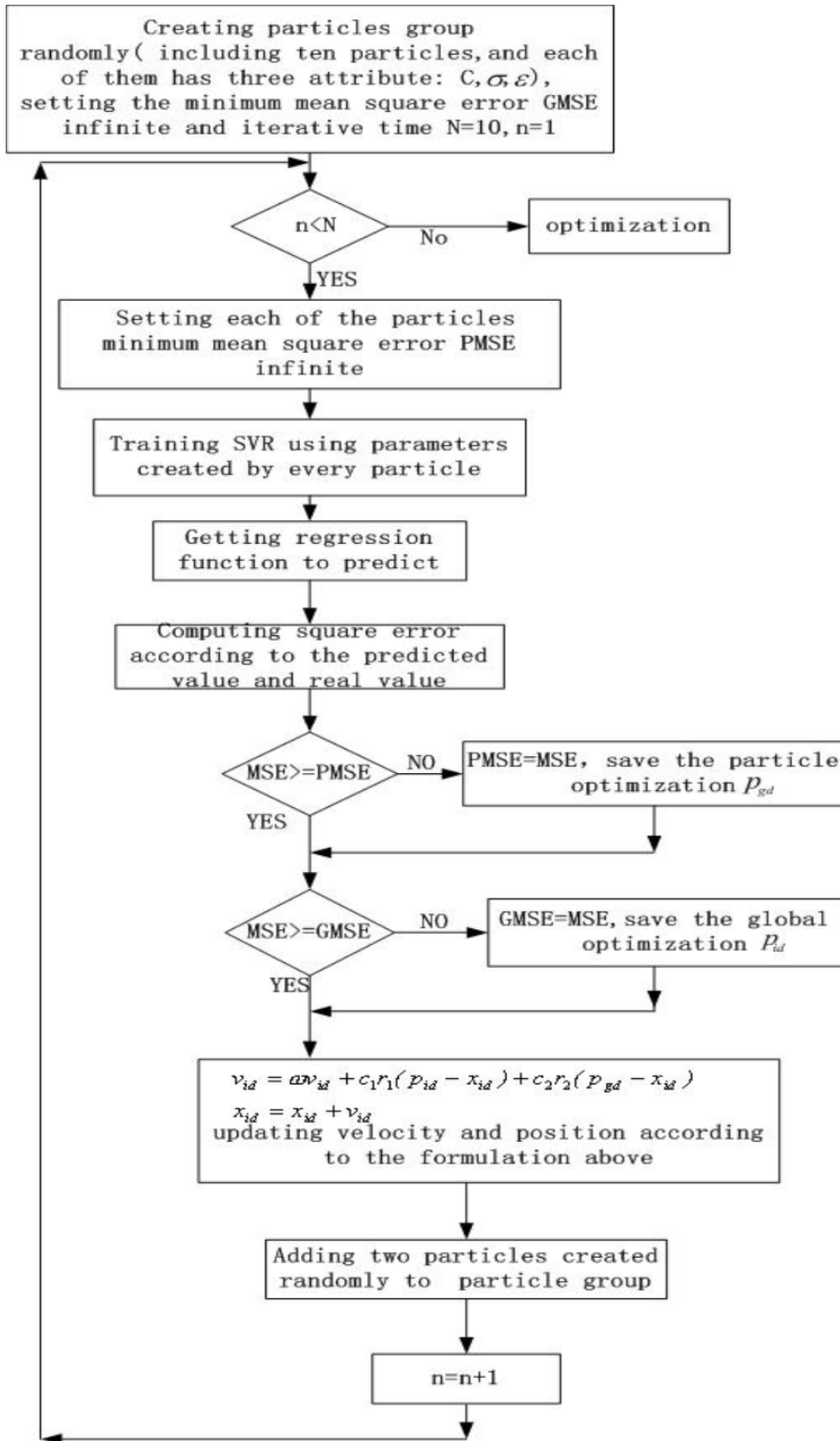


Figure 2.2 Flow Chart of the proposed method in [12]

As it mentioned before there are many studies that implement the PSO for parameter selection of SVR in real world problem such as Application of Support Vector Regression Trained by Particle Swarm Optimization in Warrant Price Prediction in [13]. It proposed a model for parameter selection of SVR based on the PSO algorithm and compared it with BP neural network (BPNN). The model used in [13] is quite similar to model introduced in [12] but with different data and application to predict warrant price that is also a very important factor to investment. The process of proposed method is exactly as follows based on [13]:

- (1) Initialize the positions and velocities of the particles.
- (2) Calculate the fitness of each particle.
- (3) Update the velocity and position of each particle.
- (4) The process proceeds until predefined number of iterations.

The warrant price data contain seven data points of absolute warrant. To train model and find best parameters fo SVR, first five data points were selected and for test the model other two warrant price data were used. Prediction error was the criteria to compare proposed model in [13] and BPNN. According to results of [13] proposed model give better performance than BPNN for this problem.

Another application of the PSO for parameter selection of SVR is studied in [23]. Guosheng et al.[23] made a model for prediction of grid resources based on SVR and the PSO algorithm. Prediction of grid resources is a critical issue for grid scheduler. In this study, performance of proposed method by [23] (SVR parameter selection with the PSO), back-propagation neural network (BPNN) and the traditional SVR model. In the traditional SVR model parameters are selected by trial and error. The PSO algorithm that used in [23] is based on [53] that it is a midified version of early PSO algorithm in [54]. In [53] sometimes velocity pass the solution area, to avoid mentioned problem, the velocity is constrained by maximum and minimum boundaries as it mentioned in [55]. Also [23] used a linear  $\omega$  as equation (7):

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \quad (7)$$

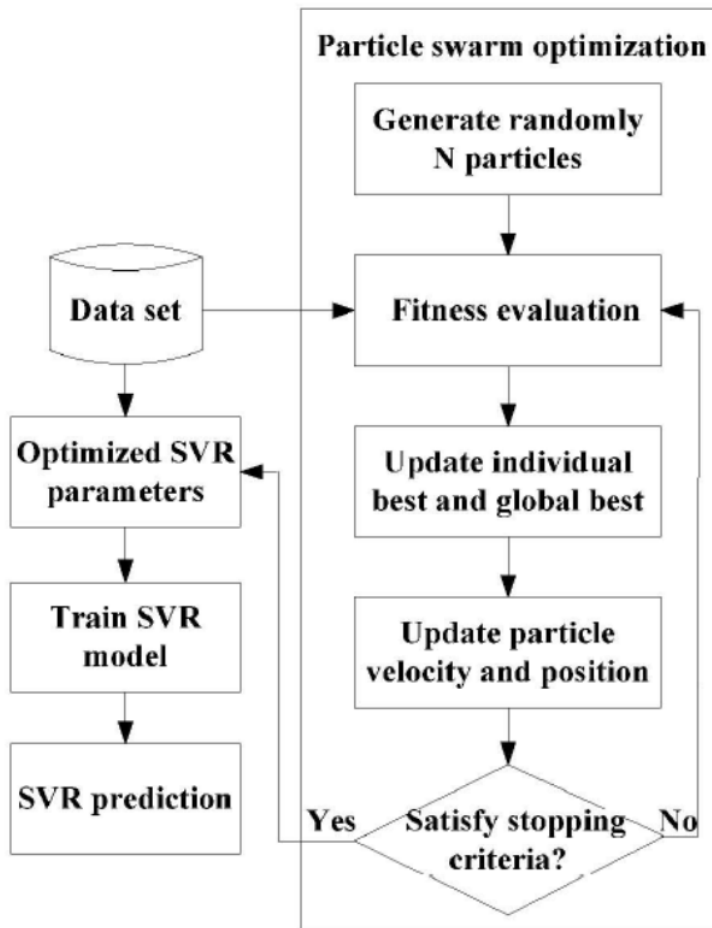
Where

$w_{\min}$  minimal inertia weight

$w_{\max}$  maximal inertia weight

$iter$  is current iteration number

Flowchart of proposed method in [23] is exactly as Figure 2.3:



**Figure 2.3** Flowchart of proposed method in [23]

Each particle is a 3-dimensional vector for optimizing  $C$ ,  $\sigma$  and  $\varepsilon$ , that are three SVR parameters must be optimized. In the fitness definition it used k-fold cross validation which  $k$  is 5. The performance of each parameter set is evaluated with Mean Square Error (MSE) which is calculating from the equation (8):

$$MSE_{cv} = \frac{\sum_{i=1}^l (a_i - p_i)^2}{l} \quad (8)$$

$a$  is a actual value,  $p$  is a predicted value and maximum number of iterations is 100.

Data used in mentioned study obtained from [56] (host load data). 200 items taken from the mentioned dataset in [56], 150 items for training and 50 for prediction. The PSO parameters was set as follow: population of particles is 20, maximum inertia weight is 0.9 and minimum inertia weight is 0.4, acceleration coefficients ( $C_1, C_2$ ) are 2, maximum velocity is 148,148,0.5 and Iterations number is 100. In conclusion results shows that PSO-SVR have better performance in comparison with BPNN and traditional SVR parater selection.

Shian and Lingzhi [57] applied support vector regression based on the PSO algorithm model to predict Rainfall. It is used support vector regression based on [58]. It is used Gaussian kernel function for SVR and parameters that must be optimized are as follows:

- i. Regularization parameter  $C$
- ii. Bandwidth of the kernel function ( $\sigma^2$ )
- iii. The tube size of e-insensitive loss function ( $\epsilon$ )

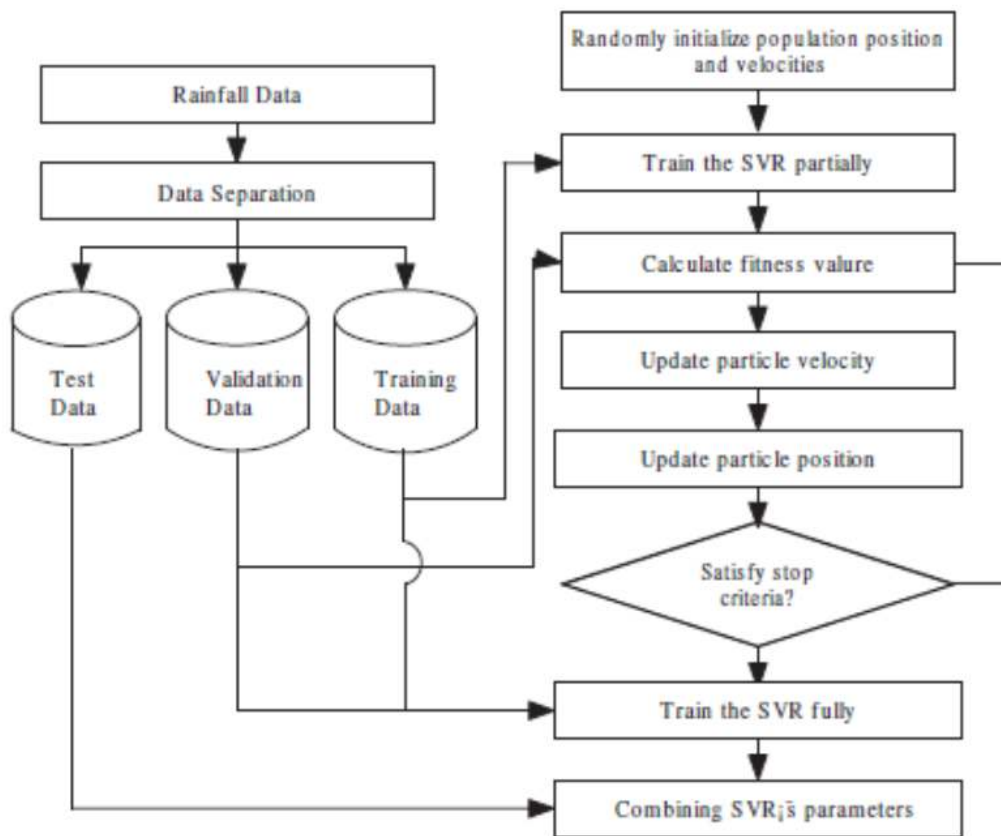
RMSE (root mean square error) is used to measure performance of the parameter set. RMSE calculated from equation (9) :

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - d_i)^2} \quad (9)$$

The fitness is calculated from equation (10):

$$fitness = \frac{R_2}{R_1 + R_2} \quad (10)$$

Where  $R_1$  is the RMSE of the training data, and  $R_2$  is the RMSE of the validation data. Flowchart of proposed method in this paper is demonstrated in Figure 2.4:



**Figure 2.4** Flowchart of proposed method in [57]

In this paper dataset is divided into three different part, one for training SVR, other for validating and the last for testing model. Dataset is used for this paper contains rainfall data of June from 1954 to 2008 at ten of Guangxi observing stations. There are 45 training samples and 10 testing samples. Also there 12 variables. This paper evaluates performance of method based on the mean absolute percentage error (MAPE), the root mean squares error (RMSE), the mean absolute error

(MAE) and pearson relative coefficient (PR). The PSO parameters used in the paper are as follows:

- i. Iteration times 100
- ii. Population size 60
- iii. The minimum inertia weight 0.1
- iv. The maximum inertia weight 0.9
- v. The minimum velocity 0.1

- vi. The maximum velocity 0.9
- vii. Learning rate 2.0

The results show that PSO-SVR have better performance over other models mentioned in the paper.

Another interesting study has been done recently is a forecasting model based on least squares support vector machine (LS-SVM) regression and the PSO algorithm on dissolved gases in oil-filled power transformers which is represented in [59]. In this model, LS-SVM regression with radial basis function (RBF) kernel and for optimization of LS-SVM hyperparameters, the PSO algorithm is used. Otherwise, four models based on back propagation neural network (BPNN), radial basis function neural network (RBFNN), generalized regression neural network (GRNN) and support vector regression (SVR), are used for comparison with proposed model in the [59]. The aim of this paper is early fault detection of power transformers which are one of the most necessary and expensive equipments in the power system.

The PSO algorithm used here is modified PSO mentioned in [53] and inertia weight is calculated by equation(3) in this chapter. Fitness value of each correspondent particle is calculated based on cross validation and it is just computed from equation (11):

$$Fitness = \frac{1}{n} \sum_{i=1}^n \sqrt{\frac{1}{m} \sum_{j=1}^m (f(x_{ij}) - y_{ij})^2} \quad (11)$$

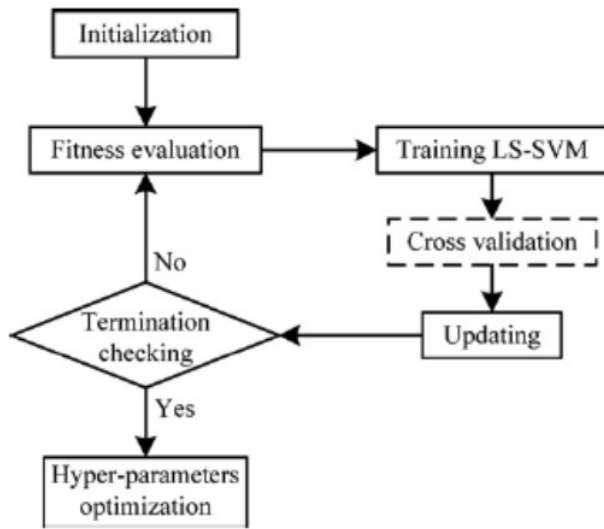
$n$  is the number of folds for cross validation

$m$  is the number of each subset as validation

$Y_{ij}$  is the true value

$f(x_{ij})$  is the forecasting value of validation samples

Flowchart of the PSO hyper-parameter optimization algorithm proposed in [59] is exactly as Figure 2.5:



**Figure 2.5** Flowchart of proposed model in [59]

The summarized procedure is as follow:

Step 1: Data preprocessing

The raw data obtained from electric power companies must be preprocessed before using it in the model for training and testing because data is sampled periodically. Raw data can transform into equal interval timeseries by interpolation methods. In this paper, cubic Hermite spline interpolation method is used to transform raw data. After transformation both training and testing data are normalized to improve generalization of LS-SVM regression.

Step 2: Hyper-parameter optimization

The PSO for hyper-parameter optimization in this paper is as follow substeps.

Step 2.1: Initialize PSO parameters also velocity is restricted to  $[-v_{max}, v_{max}]$  range which value of  $v_{max}$  defined according to experimental data.  $C$  and  $\sigma$  parameters of the LS-SVM regression are two dimensions of the each vector of particles.

Step 2.2: Calculate fitness according to equation (4) and set the best position in the swarm.

Step 2.3: For every particle train LS-SVM regression with corresponding hyper-parameters based on cross validation.

Step 2.4: Update the velocity and position of particles via (1) and (2) equations. Also update the inertia weight by equation (3).

Step 2.5: If stopping criteria is reached go to next step and if not go to Step 2.2.

Step 2.6: Termination of the algorithm and show the optimal hyperparameters.

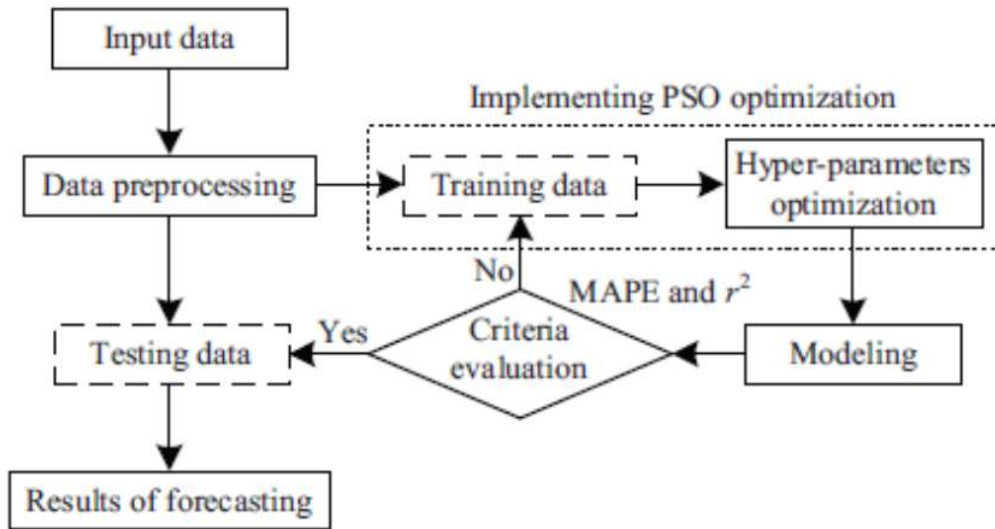
Step 3: Training and testing

Train LS-SVM with optimal hyperparameters from Step 2 to compute performance of regression model.

In this paper mean absolute error (MAPE) is used to validation of regression model which is calculated from equation (12).

$$MAPE = \frac{1}{l} \sum_{i=1}^l \left| \frac{f(x_i) - y_i}{y_i} \right| \quad (12)$$

5-fold cross validation is used to evaluate LS-SVM regression model. Population size for the PSO is 20. Maximum number of generations is 100 and initial value of inertia weight is 0.9 and at the minimum is 0.1. Flowchart of the model mentioned before is exactly as Figure 2.6:



**Figure 2.6** Flowchart of proposed model in [59]

Comparisons with BPNN, RBFNN, GRNN, and SVR models shows that the proposed method has better performance.

### 2.2.2. Feature selection based on PSO

There are two major types of feature selection: one is traditional ways and two is based on evolutionary computation algorithm. In this part of study, The PSO based cases are studied. Almost all of feature selection studies are based on the Binary PSO that some of them are illustrated briefly in the following parts.

In the image recognition field for detection of stored-grain insects, feature subset selection is one of the key issues of preprocessing part of model. Hongtao and Hanping [16] proposed a model based on the PSO algorithm and SVM for classification of stored-grained insects to improve classification accuracy. Improving classification accuracy is obtained by optimal feature subset selection. In training part of model, cross validation method was used and there are 17 morphological features in dataset used for training and testing model. At the end, results of PSO-SVM model for feature selection are compared with genetic algorithm (GA) that shows better performance upon GA. In mentioned paper [16], binary PSO (BPSO) and MULTI-CLASS SVM are used to build a model.

There are two criteria due to devise a fitness function: first is classification accuracy and second is number of selected features. The idea behind this, is to create a single fitness function that cover two goals. In other words, if two particles have a same accuracy or number of features, the one with higher accuracy or lower number of features has a higher fitness value. The value of fitness function is defined by equation (13):

$$F = P_{svm} * 100 - d/l \quad (13)$$

$d$  is the number of the selected features

$l$  is the number of the all features

$P_{svm}$  is the  $v$ -fold cross-validation

Each dimension of the particle's position is represented by one binary value, 1 means that dimension is selected and 0 means it is not selected. Population size is 50, number of dimensions are 17, maximum iteration is 20, the

mutation probability of the particle bit is 0.3, the inertia weight  $\omega$  decreases from 0.9 to 0.4 linearly, the penalty factor  $C=20$ , SVM uses RBF as kernel function and the RBF kernel parameter  $g$  is  $10/d$ . Procedure of the model in [16] is exactly as follows:

Step 1. Particle initialization and PSO parameters setting:

Generate initial particles. Set the PSO parameters including number of iterations, velocity limitation, number of particles, particle dimension. Set iteration = 0, and perform the training process from step 3–8.

Step 2. Set iteration  $k = k + 1$ .

Step 3. SVM model training:

(a) Training and validation set preprocessing: select input features for training and validation data sets according to the particle.

(b) SVM classifier accuracy calculation: for the training set, conduct v-fold cross validation and calculate the average cross-validation accuracy.

Step 4. Fitness evaluation: For each particle, evaluate its fitness by formula (5).

Step 5. Update the global and personal best according to the fitness evaluation results. Record the average training crossvalidation accuracy for the global and personal best.

Step 6. Update particles position.

Step 7. Particle mutation: avoid the search getting trapped in a local optimal point, and introduce the mutation operator from GA, mutate particles except the global best particle.

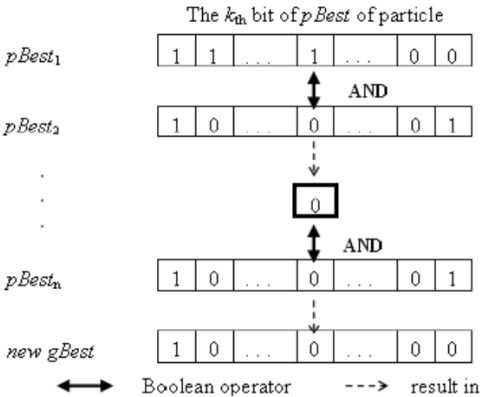
Step 8. Stop condition checking: If the maximum iteration is met, go to step 3, otherwise, go to the next step.

Step 9. Output the optimal feature subset, measure testing accuracy on test set via the trained SVM classifier.

135 samples are generated for training and 90 for testing. According to [16] results of BPSO-SVM are better than GA and BPSO-SVM accuracy is satisfying and also it is upon 90%.

Cheng-San et al.[15] proposed a new version of binary particle swarm intelligence called Boolean binary particle swarm optimization for feature selection. This paper introduced a boolean function to overcome disadvantages of the standard particle swarm optimization. Results of subset selection that provided by this method was used in classification. Several microarray datasets were used to test mentioned method.

In boolean binary particle swarm optimization, at first positions and velocity of particles started randomly. There is a gbest particle which is the fittest particle in the swarm. Each particle is affected by its own pbest and gbest. If gbest is remain unchanged, all particles gather around gbest. However, clustering particles around gbest cause useless computation results in the next generations. In the other word, the algorithm stick in a local optimal and may be it can not pass that local optimal position. Boolean binary particle swarm optimization function introduced to overcome the mentioned problem. The basic boolean function have three basic operators AND, OR and NOT. In BPSO, after some iteration, if classification accuracy remain unchanged, the algorithm suppose that the algorithm trapped in a local optimal. In B-BPSO, if classification accuracy remain unchanged after three generations, the algorithm generate a new position to gbest that influence all particles. After assigning a new position to gbest, all particles depart from their location and go to search other areas of search space rather than trap in a position. Due to achieving mentioned goal, AND(.) operator used to ‘and’ every bit of all pbest of particles. The following figure (Figure 2.7) shows the process of and operation.



**Figure 2.7** Representation of the boolean operation to reset global best based on [15]

The Pseudo code for assigning new gbest is as follow:

```

Begin
For d=1 to number of dimension of particle
  For p=1 to number of particles-1
    New gbestd=original gbestd ‘AND’ original gbestd+1
  End for
End for

```

*End for*

*end*

Also the pseudo-code of the procedure of B-BPSO based on [15] is as follows.

*Begin*

*Randomly initialize particle swarm*

*While (Numberof iterations, or the termination criterion is not met)*

*Evaluate fitness of particle swarm*

*For p=1 to number of particles*

*If fitness of  $X_p$  is greater than fitness of  $pbest_p$  then*

*$pbest_p = X_p$*

*end if*

*if fitness of the particle in swarm is greater than  $gbest$  then*

*$gbest = \text{position of particle}$*

*end if*

*for d=1 to number of dimension of particle*

*$v_{pd}^{new} = w \cdot v_{pd}^{old} + c_1 \cdot rand_1 \cdot (pBest_{pd} + x_{pd}^{old}) + c_2 \cdot rand_2 \cdot (gbest_d - x_{pd}^{old})$*

*If  $v_{pd}^{new} \notin (V_{max}, V_{min})$  then  $= \max(\min(V_{max}, v_{pd}^{new}), V_{min})$*

*End if*

*Next p*

*If  $gbest$  fitness value is identical after three iterations*

*Create new  $gbest$*

*Using new  $gbest$  replace original  $gbest$*

*End if*

*Next generation until termination criterion*

*End*

Data used in [15] is obtained from <http://gems-system.org> that is about cancer related human gene expression. It used six cancer related human gen expression datasets with different number of classes and samples.

B-BPSO provided feature subset were used with K nearest neighbor classification algorithm and also k-fold crossvalidation is used to evaluate the results. The results of B-BPSO were compared with BPSO and GA. In this paper number of particles was 30, acceleration factors  $C_1=C_2=2$ , the inertia weight  $w$  is 1, and maximum number of generations was 100. For the GA, population size is 30, crossover rate is 1 and mutation

rate is 0.1. Experiments show that results of B-BPSO are better than other two (BPSO and GA).

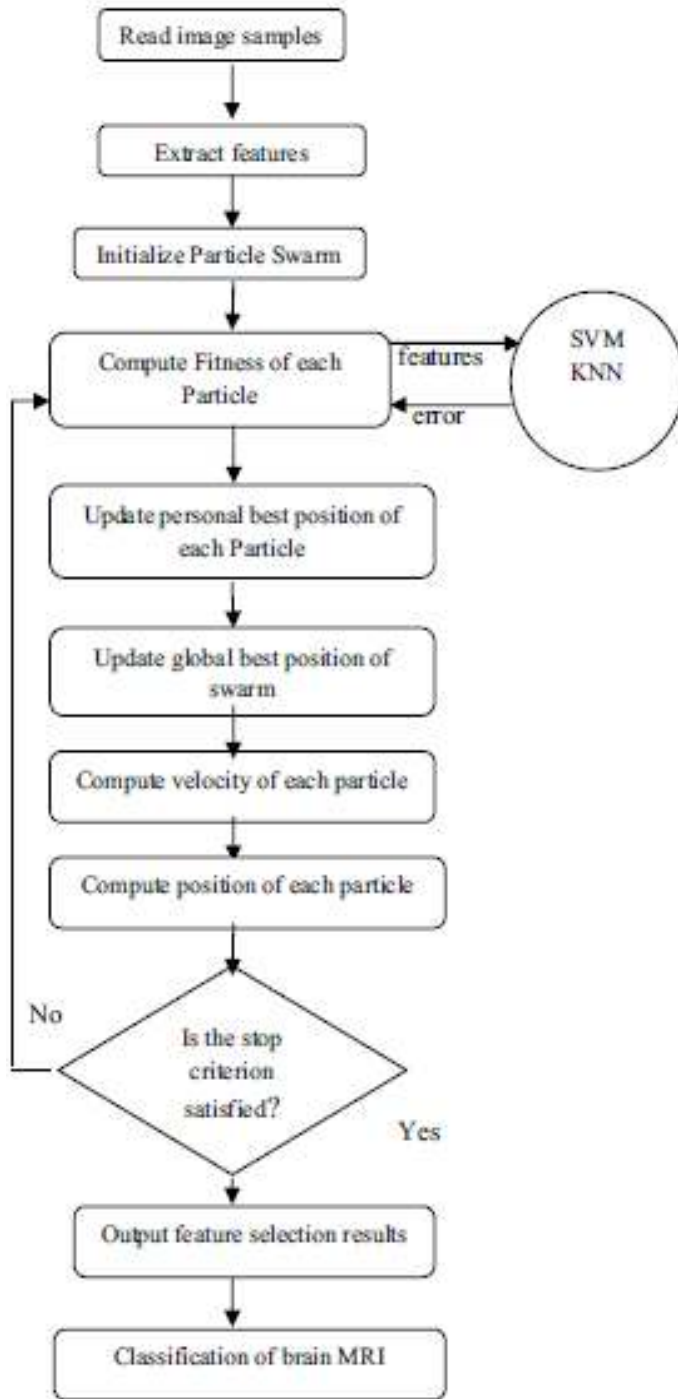
In diagnostic medicine field, automatic classification of Brain MRI (magnetic resonance imaging) is one of the interesting study areas that many studies have been done. In this field, first of all, all needed features must be extracted from images via image processing techniques and after that, those features are used in classification techniques to classify normal and abnormal brains and also may be for other purposes such as classification of tumors and etc.. In the mentioned field, Feature subset selection is a critical issue and has a great impact on the performance of classification. Rehman et al. [60] proposed a hybrid method for Feature Selection and Tumor Identification in Brain MRI using Swarm Intelligence. In [60], Discrete Binary Particle Swarm Optimization (DBPSO) is used as a popular version of the PSO for feature subset selection. Discrete binary PSO is employed to combination optimization problems. SVM and K-Nearest Neighbor (KNN) are used to classify each record into normal or abnormal categories. Experimental results show that the proposed approach reduces the number of features and at the same time it achieves a high accuracy level. PSO-SVM is observed to achieve a high accuracy level using a minimum number of selected features [60].

In order to extract features from images, GLCM method is used that is presented by [61]. The GLCM is proved to be an effective and efficient procedure for providing second order statistical texture features. In this field, number of features are high and reduction number of features to achieve greater accuracy is an important issue.

In DBPSO like BPSO, each dimension of a particle represents with bit value which can be 1 or 0. If 1, it means that feature is selected otherwise not. The details of the DBPSO are according to [62]. The procedure of this paper is quite similar to the basic PSO mentioned before so it is not necessary to write it down here. Two different hybrid techniques based on DBPSO are devised, DBPSO-SVM and DBPSO-KNN that are implications of the DBPSO with two different classifiers. Flowchart of mentioned method according to [60] is exactly as Figure 2.8.

Data set used for this purpose contained 20 T2 weighted Brain MRI images taken from MRI Center which 6 of them belong to normal images and other 14 belong to abnormal images. For testing classification accuracy 2 fold cross validation is applied. Also wavelet

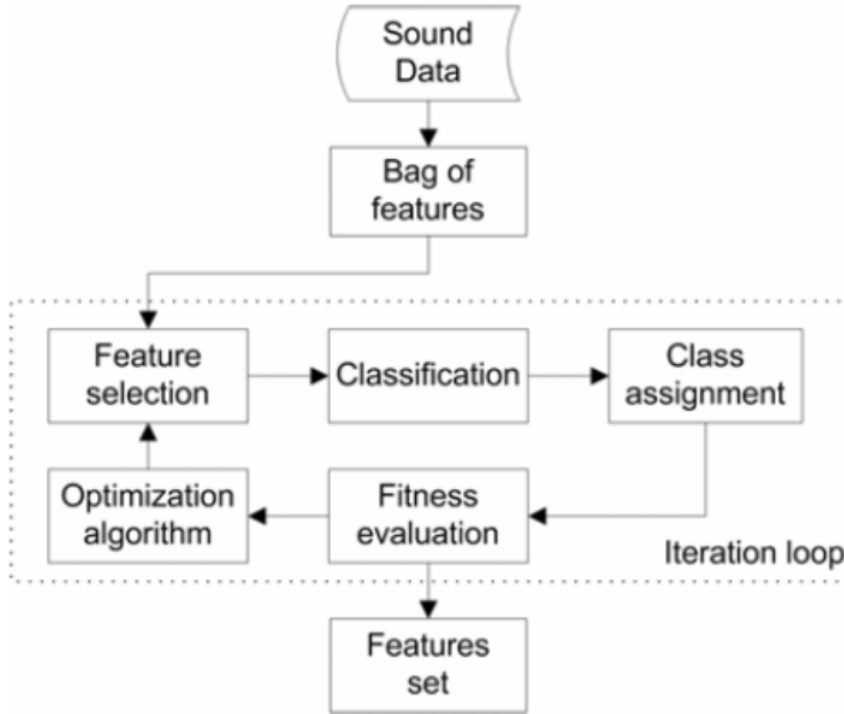
feature extraction is used to extract features. Total number features is 78, 66 GLCM based and 12 wavelet based features. Different kernel types used for SVM are linear Kernel, polynomial degree 2, polynomial degree 3, polynomial degree 4, polynomial degree 5, and Radial Basis Function (RBF). In the term of K for K-NN three values used are K=3, K=5 and K=7. Based on outcome represented by this paper, SVM polynomial degree 5 kernel has the best accuracy and K=3 for K-NN shows the best found accuracy in their own category. The DBPSO parameters were, swarm size or population size is 100, maximum iteration number is 100,  $w=0.5$  and  $c1 = c2 = 2$ .



**Figure 2.8** Flowchart of the model built in [60]

Chmulik et al.[63] studied the PSO and GA for affective sounds discrimination problem. For objective acoustic feature subset selection for automatic sound detection, the PSO and GA are used in this paper. For this purpose, data set contains two hours of sounds

such as cry, laughter and applause, also several hours of other sounds tagged to the data set such as speech, music and noise. There is 4-9 % performance increasing and between 50 and 90% decreasing in feature space. Paper used standard PSO algorithm based on [41] and it is explained before in this chapter. The flowchart of the procedure according to [63] is as Figure 2.9.



**Figure 2.9** Flowchart of the model built in [63]

Applied GA algorithm was used based on [64]. In this paper experiments are based on bi-modal classification problem, which classifier make a decision to classify a record to a class or not. The K-NN classifier used to experiment in mentioned paper. Evaluation of fitness is based on classification accuracy and number of selected features. The data set contain six semantic classes cry, laughter, applause, speech, music and noise. After signal parameterization and feature extraction, the results go to bag of features that consist of 20 features.

Euclidean distance was used in the K-NN algorithm and  $K=7$ . In the case of the PSO, Maximum iteration number is 100, number of particles is 25, learning factors  $c_1 = c_2 = 1.495$ , maximal velocity  $v = 4$  and inertia factor  $w = 0.7284$ . For GA, population size is 25, 2 points crossover with crossover probability  $pC = 0.85$  and mutation probability

$pM = 0.05$ . Every optimisation process took 50 iterations. The standard F measure (harmonic mean of precision and recall) was used to show classification accuracy. The fitness value is outcome of combination of three different values, clip level classification accuracy, frame level classification accuracy and the number of descriptors (solutions with less features were preferred among the others).

### 2.2.3. Simultaneous parameter and feature selection based on PSO

Early studies focus on feature subset selection or Parameter optimization for SVM separately but in the last years, there are papers that studied simultaneous feature selection and parameter optimization. Many researchers used basic form or standard version of the PSO but due to overcome disadvantages of the basic PSO (continuous and binary), many studiees proposed modified version of the PSO or BPSO for simultaneous feature selection and parameter optimization. In this part of study, we discuss some of these papers that applied to various problems.

Susana et al.[42] proposed a modified binary version of the PSO (MBPSO) for feature selection with SVM kernel parameter optimization for mortality prediction of septic patients. The aim of proposing a modified version of the BPSO in mentioned paper is to tackle premature convergence of the BPSO algorithm. There are two class of (survived or deceased) of patients with septic shock. Several benchmark datasets and other versions of the PSO with GA are used to test MBPSO. Based on the outcome of experiments in [42] results showed that the proposed method can achieve high classification accuracy and select better feature subset in comparison with other PSO based algorithm, but in the case of GA, results are similar but there are less features in MBPSO based results than GA.

In this paper, the aim of feature selection is to improve quality of modeling. Classification used to evaluate feature selection quality. There are two goals is mentioned study and they are maximization of model performance and the minimization of the number of features used in classification. Based on [42] the most suitable objective function or fitness function is equation (14) based on [65]:

$$f(u_i) = \alpha(1 - P) + (1 - \alpha) \left( 1 - \frac{N_f}{N_i} \right) \quad (14)$$

P is the classifier performance measure

$N_f$  is the size of the feature subset

$\alpha \in [0, 1]$  define the weights of performance and subset size

Accuracy of the classification computed according to equation (15).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (15)$$

TP is true positives

TN is true negatives

FP is false positives

FN is false negatives

There are two problems with accuracy. If one class is in a minority, misclassification of that class has not enough influence in the value of the accuracy. Acceptable classification must classify other classes too. These two mentioned issues cannot be addressed with accuracy. In order to address this problem, additional performance measures were considered by the authors according to [66] are compute from equations (16) and (17).

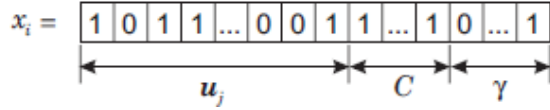
$$Sensitivity = \frac{TP}{TP + FN} \quad (16)$$

$$Specificity = \frac{TN}{FP + TN} \quad (17)$$

Sensitivity is equal to the TP rate and specificity is equal to the TN rate.

The tendency to prematurely converge is a serious problem in dealing with BPSO that mentioned in [67]. There are several studies address this problem and proposed solutions such as implement a mutation operator [36], reinitialize best particle in the swarm [37], using perturbation mechanisms [68]. Proposed method by [42], addressed mentioned problem and took benefits of papers in [36,37,68]. The method studied in [42] also studied in [69]. The SVM classifier with RBF kernel function was used with two parameter to optimize, C and  $\gamma$ . In this way, RBF function's parameters(C and  $\gamma$ ) and features altogether form a particle. Each particle has three part, two parts for SVM

parameters and third part is binary sequence of bits refers to the feature selection. Also binary values used for parameter optimization must decode to real values before use. Representation of a particle according to [42] is as Figure 2.10.



**Figure 2.10** Representation of a particle in [42]

Modified binary particle swarm optimization (MBPSO) is a mixture of reset best particle of swarm and local search operator that is similar to mutation. In the MBPSO local search operator, changing in the particles influence the value of Vmax instead of using probability. This technique, first mentioned in [70] then it used in MBPSO. In MBPSO, if in a constant number of iteration gbest value stay unchanged it must be reset which usually is small number (it is recommended three in [15]) to avoid convergence of all particle to gbest particle. Local search in MBPSO displace pbest values of particles according to the equation (17):

$$\begin{cases} x_{ij}^{pb} = -x_{ij}^{pb} & \text{if } r \leq dr \\ x_{ij}^{pb} = x_{ij}^{pb} & \text{otherwise} \end{cases}, i=1, \dots, N, j=1, \dots, N_p \quad (17)$$

dr is the displacement rate

r is a random number  $\in [0, 1]$

N is number of dimensions

$N_p$  is number of particles or swarm size

MBPSO uses a mechanism that make some small “mistakes” while updating particle position. This mechanism is similar to the mutations in EA, for avoiding premature convergence.

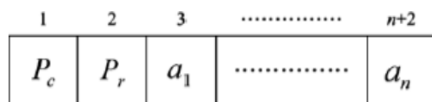
Six data set used for testing MBPSO demonstrated in Table 2.2.

**Table 2.2** Datasets and their characteristics used to evaluated model in [42]

Dataset Name	Samples	Features	Number of classes
German (credit card)	1000	24	2
Sonar	208	60	2
WBCO	683 (699)	9	2
WPBC	198	32	2
WDBC	569	30	2
Colon cancer	62	2000	2

The benchmark datasets used here were obtained from UCI data repository in [71]. MBPSO were compared with BPSO, IBPSO [37] and GA [72]. Also 10 fold cross validation applied in the MBPSO as validation mechanism. Parameters of the MBPSO were selected according to [15] that discussed in feature selection literature review. Finally results of experiments showed that this method could achieve better accuracy over other algorithms used.

There is another paper that studied simultaneous parameter determination and feature subset selection. Shih-Wei et al.[17] too, the SVM classification used to evaluate performance of model. In the other word, performance of model, compute based on classification accuracy. Various public datasets were used to evaluate proposed model by this paper. Results were compared with grid search and GA. Proposed model in [17] are better than grid search but similar to GA. The PSO algorithm is a modified version of PSO based on [53]. In this study, values of each dimension of particles are continuous values unlike most of studies that used binary values. There are two parameters of the SVM must be optimized, C and  $\gamma$ . Each correspondent dimension of a particle to a feature has a value between 0 and 1. If the value is equal or less than 0.5 then that feature is not selected otherwise when the value is greater than 0.5, the correspondent feature is selected. Representation of each particle is demonstrated in Figure 2.11.



**Figure 2.11** Representation of a particle in [17]

$$P_c = C$$

$$P_r = \gamma$$

$a_i$  is a dimension to a feature

The general process of the PSO is quite similar to one that mentioned in the “Basic PSO Algorithm Pseudo Code” early in this chapter. There are 17 data sets used to test model that taken from UCI data repository [71]. These 17 data set different number of features (from 4 to 60) and also samples (150 to 1728). K-fold cross validation used too which value of K is 10. For each test program run 10 times. Accuracy was computed by averaging of accuracies for 10 runs. Architecture of the model proposed in [17] is exactly as Figure 2.12:

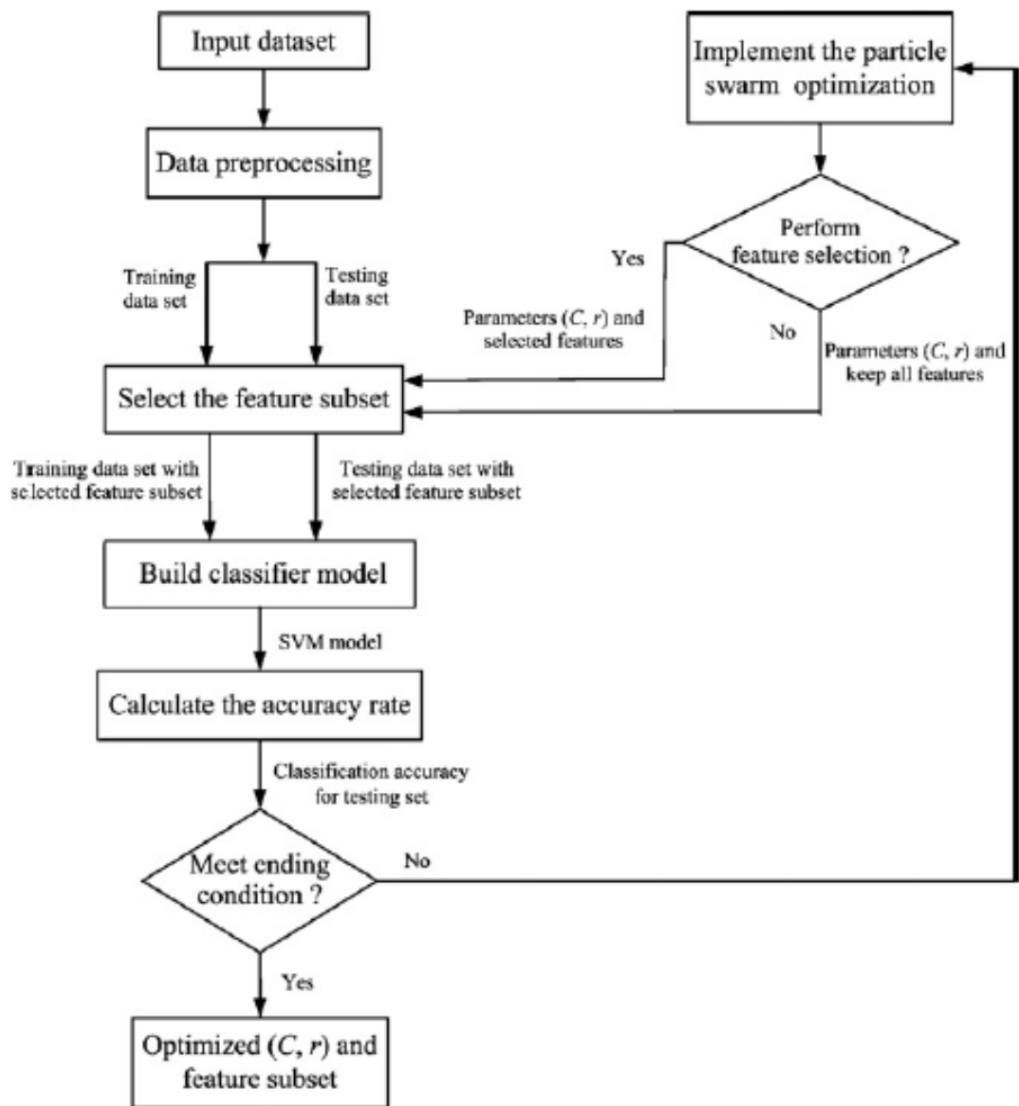


Figure 2.12 Architecture of the model proposed in [17]

The PSO parameters have been set as follow:  $C_1=C_2=2$ , in the case of just SVM kernel parameter optimization, number of particles set to 6 and maximum number of iterations set to 50. But, in the case of simultaneous feature selection and parameter optimization, number of particles set to 8 and maximum number of iterations set to 250. Search rang of the SVM parameter  $C$  was between 0.01 and 35,000 and  $\gamma$  was between 0.0001 and 32. There are two models in this paper: The one proposed based on the PSO and the one based on GA. Each model tested with and without feature selection. In the other words, each model test for just SVM parameter optimization and simultaneous parameter optimization and feature selection. Results showed that results with simultaneous parameter optimization and feature selection are better than just parameter optimization for but models. But outcome of PSO-SVM and GA-SVM are similar and there is no significant difference between them.

Another application of the PSO and SVM is in short-term load forecasting (STLF). Ying-Chun [39] proposed a hybrid method based on the PSO and SVR for short-term load forecasting that feature selection and parameter determination for SVR were done simultaneously. Proposed method compared with PSO-SVR without feature selection and the traditional SVR. The average process time and the forecasting accuracy for the proposed method were better than PSO-SVR without feature selection and the traditional SVR.

RBF kernel function used for SVR, because of its ability to analyze higher dimensions and also there are three parameters need to optimized,  $C$ ,  $\sigma$  and  $\varepsilon$ . Each particle represented as Figure 2.13.

(discrete) Input feature mask	(continuous)		
	$C$	$\sigma$	$\varepsilon$
$x_{i,1}, x_{i,2}, \dots, x_{i,nF}$	$x_{i,nF+1}$	$x_{i,nF+2}$	$x_{i,nF+3}$

**Figure 2.13** Representation of a particle in [39]

In this paper, each particle has two parts: First, a sequence of discrete values for feature selection and second, three continuous values for SVR hyper parameters determination. First part just like BPSO method that mention before, if value is 1, it means that feature is selected, otherwise, if value is 0, it means that value is not selected. There are two criteria involved in design of fitness function: prediction error and number of features

that selected. Therefore, particles with less features and higher accuracy produce lower error. The aim of this devise is to provide one objective function with multiple criteria. Equation (18) shows the objective function used in [39].

$$error_i = w_A \times MAPE_i + w_F \times \frac{\sum_{j=1}^{n_f} f_j}{n_f} \quad (18)$$

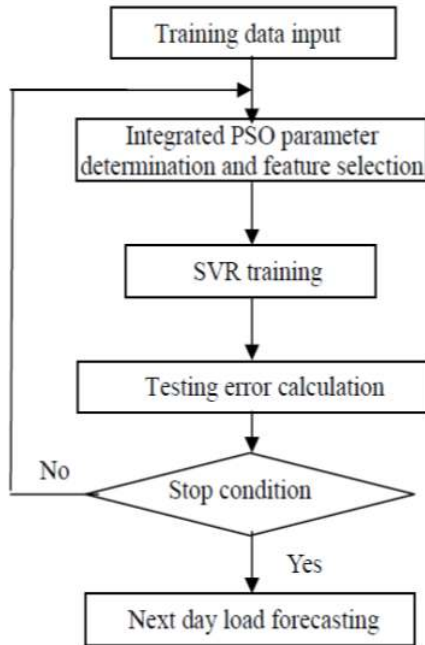
Where  $w_A$  is the prediction accuracy

$w_F$  is the selected feature

$f_j$  is the value of feature mask, if value is “1” represents that feature  $j$  is selected and “0” represents that feature  $j$  is not selected

$n_F$  is the total number of features.

K-fold cross validation used as validation method. The general process of proposed method by [39] is exactly as Figure 2.14:



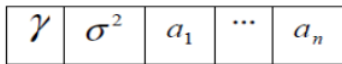
**Figure 2.14** General process of proposed model by [39]

Short-term load forecasting (STLF) could be treated as a regression problem and it is in the multivariate category. The outcome of this model is the load of next day. There are several features such as average temperature ( $T_a$ ), highest temperature ( $T_h$ ), lowest

temperature (Tl), humidity index (H), wind power (Wp) , type of the day D(d), type of the week W(d) and the rain fall (Rf) etc. Data provided from electric system from 2006 to 2007 in China. There are 24 load points for each day and correspondent features of each day. The goal is to predict 24 load points of next day based on historical data.

Each particle has 11 dimensions, 3 for SVR and 8 features. Swarm size is 40 and maximum generations' number is 100. The range of SVR kernel parameters that to be search are  $C \in [0.01, 1000]$ ,  $\sigma \in [0.01, 10]$ , and  $\varepsilon \in [0.01, 0.9]$ .  $V_{\max}$  was about 2–10% of the dynamic range of the variable on each dimension for the continuous type of dimensions.  $C_1=C_2=2$ . The MAPE's weight  $w_A$  is adjusted to 95%, and the feature size's weight  $w_F$  is set to 5%. The simulation done by Matlab7.0. Based on experimental outcomes in [39], proposed approach by paper could select input features appropriately and in comparison with basic PSO-SVR and the traditional SVR, average process time and the forecasting error decreased.

Other approach based on the PSO and SVM is based upon [19], which used the PSO to feature selection and optimization of LS-SVM kernel parameters. Standard version of the PSO with RBF kernel of LS-SVM used to build model. The proposed model compared with GALS-SVM and LS-SVM. Various UCI datasets used evaluate model. Also in [19] is a classification problem. Analysis of results revealed that PSOLS-SVM outperformed other two methods. In this paper BPSO with linearly decreasing inertia weight was applied to feature selection and parameter optimization of LS-SVM. RBF kernel choose as SVM kernel function because in [73] mentioned that it has better performance over other kernel functions. There are two parameters of LS-SVM needed to be determine:  $\gamma$  and  $\sigma^2$ . So, each particle contains two parameters and some features according to data set used. Figure 2.15 shows representation of particle.



**Figure 2.15** Representation of a particle in [19]

If value of corresponding bit is“1” represents that feature is selected and, if “0” represents that feature is not selected.  $\gamma$  and  $\sigma^2$  must be decode to a real value before use. Fitness function calculated from equation (19):

$$fitness = \frac{1}{RMSE(\sigma^2, \gamma)} \quad (19)$$

Where, RMSE calculate from formula (20).

$$RMSE = \sqrt{\frac{1}{N} \sum_{k=1}^N e_k^2} \quad (20)$$

Smaller fitness value means stronger adaptiveness of the model. Breast Cancer, Cleveland Heart, German, Sonar and Wine Data sets from UCI machine learning Library were used to evaluate the model [71]. The following Table shows data set characteristics.

**Table 2.3** Characteristics of used datasets according to [19].

Dataset name	No.of classes	No.of instances	Numeric features
Breast Cancer	2	683	10
Cleveland Heart	2	296	13
German	2	1000	30
Sonar	2	208	60
Wine	3	175	13

Development environment was Visual C++ 6.0. Vmax = 50 and maximum number of iterations for the BPSO was 100. K was 5 for K-fold cross validation. For each dataset PSOLS-SVM algorithm run five times. Due to calculate accuracy of classification, algorithm summing of each accuracy at the end of every run, then divide that number by five. Number of particles in a swarm or population size were different for each dataset and they were as following: Each dataset uses the following: Cleveland Heart and Wine are 100, Breast Cancer is 50, German is 200, Sonar is 300. According to the outcome of testing, PSOLS-SVM has better results than two other models mentioned before.

Jiansheng and Enhong [18] proposed a hybridized the particle swarm optimization HPSO–SVR. This model is hybridized the particle swarm optimization (PSO) and support vector regression (SVR). The aim of this study is to improve the regression accuracy based on kernel function type and kernel parameter value determination also with feature subset selection. This model is applied to rainfall data to forecasting monthly rainfall. This model is a combination of the discrete PSO with the continuous–valued PSO which can take advantages of both of them in simultaneously optimize the

input feature subset selection, the type of kernel function and the kernel parameter determination of SVR. The test results of this model with respect to other models showed that HPSO–SVR model outperforms the previous models. HPSO–SVR performed successfully in optimal feature subset selection, optimal kernel function type and optimal kernel parameter determination of SVR in the rainfall forecasting data.

Three kernel types used in this model were as follow:

- i. Linear kernel: which has one parameter.
- ii. Polynomial kernel: which has two parameters.
- iii. RBF kernel: which has two parameters.

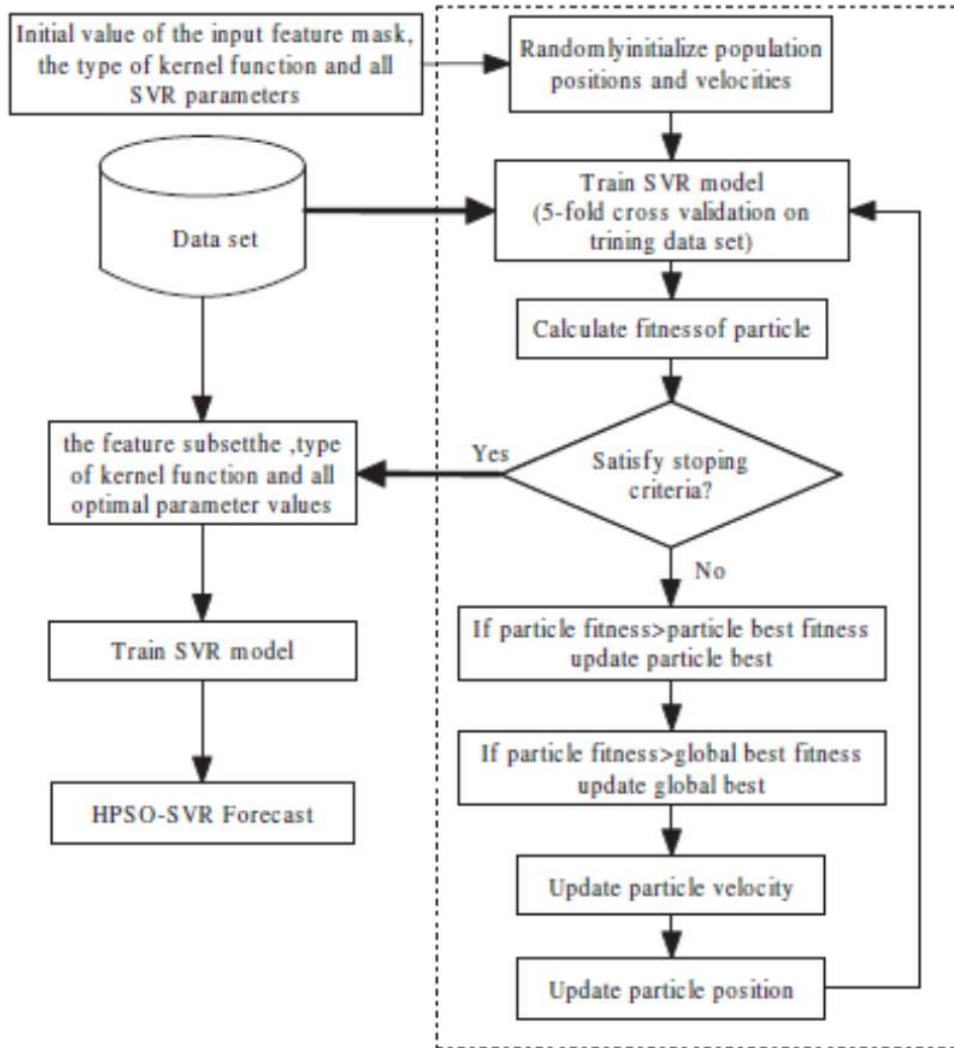
The LS–SVM used with mentioned kernel types to build a model. As it mentioned before, HPSO is a combination of the discrete PSO with the continuous value PSO. For the feature mask of discrete part of particle, if value of corresponding bit is “1” represents that feature is selected and, if “0” represents that feature is not selected. Representation of a particle according to [18] is exactly as Figure 2.16:

Input feature mask (discrete)	Kernel Type (discrete)	Parameters(continuous)
$x_1 \ x_2 \ \dots \ x_m$	$x_1 \ x_2 \ \dots \ x_n$	$x_1 \ x_2 \ \dots \ x_t$

**Figure 2.16** Representation of a particle in [18]

$x_1, x_2, \dots, x_m$  represents the feature mask, m is number of features that can vary from one dataset to another.  $x_1, x_2, \dots, x_n$  represents the type of SVR kernel, n is the number kernel types that used.  $x_1, x_2, \dots, x_t$  represents the parameter of SVR kernel must be optimize, t is number of kernel parameters.

K-fold cross validation applied to evaluate performance of the model. The performance of features and parameters set measured by mean abstract percent error (MAPE). Process of the HPSO-SVR model according to [18] is as Figure 2.17:



**Figure 2.17** Process of the HPSO-SVR model according to [18]

Development environment was Visual C++ 6.0. Maximum number of generations was 100, maximum number of iteration was 100,  $C_1=C_2=2$ , maximum inertia weight was 0.9 and minimum inertia weight was 0.1. K in the K-fold cross validation was 5. Dataset was monthly rainfall data from 1958 to 2009 that gathered from 8 stations from Guangxi. Total number of data points in dataset are 624. Training set have 564 (1958–2003) and testing set have 60 (2004–2009) samples. There are 70 variables as forecasting factors. Due to measure the performance of the HPSO-SVR, normalized mean squared error (NMSE), the mean absolute percentage error (MAPE) and the pearson relative coefficient (PR) were used.

## **2.3. Artificial Bee Colony**

### **2.3.1. Swarm intelligence and behaviour of real bees**

Artificial bee colony (ABC) algorithm is a high-performance optimization technique that introduced by Karaboga [75] in 2005. Artificial bee algorithm mimics the foraging behavior of honeybee swarm [76] and applied several engineering problems successfully[76-77]. The ABC has attracted much attention of researchers because of its simplicity or ease of implementation, robustness, fewer control parameters and superior performance. Also the mentioned reasons make it an appropriate algorithm for solving optimization problems. In literature, ABC has been successfully used for feature selection optimization [78] and [79] based on rough set theory. ABC for parameter optimization of SVM has been applied in [80] and [81] successfully. The results of ABC algorithm in both feature selection and parameter optimization fields are promising.

Swarm intelligence (SI) is the associated behavior of the individual agents. SI systems are normally made up of a population of simple agents that interacting with environment or other individual. The basic idea of SI comes from nature. Mostly there is no central control of the swarm and individuals behave with simple rules and aggregation of such a behavior cause intelligent behavior of swarm. Ant colony and bird flocking are two well-known examples of swarm intelligence.

There are many kinds of swarms in the nature that do not have intelligent behavior, also level of intelligence is different from one to another. Bonabeau et al. [82] and Millonas [83] defined some characteristics for intelligent swarms that are mentioned in the following.

Characteristics of self-organization in swarm specified by Bonabeau et al.[82] as below:

- (1) Positive feedback is a simple behavioral “rules of thumb” that promotes the creation of convenient structures. Recruitment and reinforcement such as trail laying and following in some ant species or dances in bees can be shown as examples of positive feedback.
- (2) Negative feedback counterbalances positive feedback and helps to stabilize the collective pattern. In order to avoid the saturation which might occur in terms of

available foragers, food source exhaustion, crowding or competition at the food sources, a negative feedback mechanism is needed.

(3) Fluctuations such as random walks, errors, random task switching among swarm individuals are vital for creativity and innovation. Randomness is often crucial for emergent structures since it enables the discovery of new solutions.

(4) Multiple interactions occur since agents in the swarm use the information coming from the other agents so that the information and data spread to all network.

Millonas [83] defined rules must be satisfied by each swarm to be categories as an intelligent swarm:

(1) The swarm must be capable of doing time and space computations that known as the proximity principle.

(2) The swarm must be able to compute quality factors of environment and do appropriate reaction based on that, which is known as quality principle.

(3) The swarm must distribute resources into as many places as possible , which is known as the principle of diverse response.

(4) The swarm must have stable behavior on every fluctuation of environment, which is known as the principle of stability.

(5) The swarm must have capability of changing behavior when the investment in energy is worth the computational price, which is known as the principle of adaptability.

Global behavior of a swarm and interaction of each individual with others have modeled by ethnologists according to characteristics described by Bonabeau et al.[82] and Millonas [83]. Recently researchers found that they can use these models to solve real world problems such as routing, networking and etc. Ant colony represented by Dorigo et al.[84] and bird flocking represented by Kennedy and Eberhart [85] are two of well-known swarm intelligence algorithms that are used to solve many problems. Also many researchers modified early versions of algorithms due to performance, trapping in local optimum and etc.

Real bees foraging behavior model have three essential components: food sources,

employed foragers, and unemployed foragers, and also contain definition of two leading modes of the honeybee colony behavior: recruitment to a food source and abandonment of a source [86]. Explanation of each component of honey bee colony is as below:

1. Food Sources: Due to choose a food source, some features or issues that are related with food source should be evaluate by a forager bee such as how much is close to beehive, enrichment of energy, quality of nectar, and how easy or difficult to extracting energy form that sources. Food source quality could be illustrate by one quantity that it depends on mentioned issues above.

2. Employed foragers: An employed forager bee is engaged at a particular food source which she is use currently. On the other side for bees that are waiting in beehive she take some information such as distance, direction and the profitability of the food source.

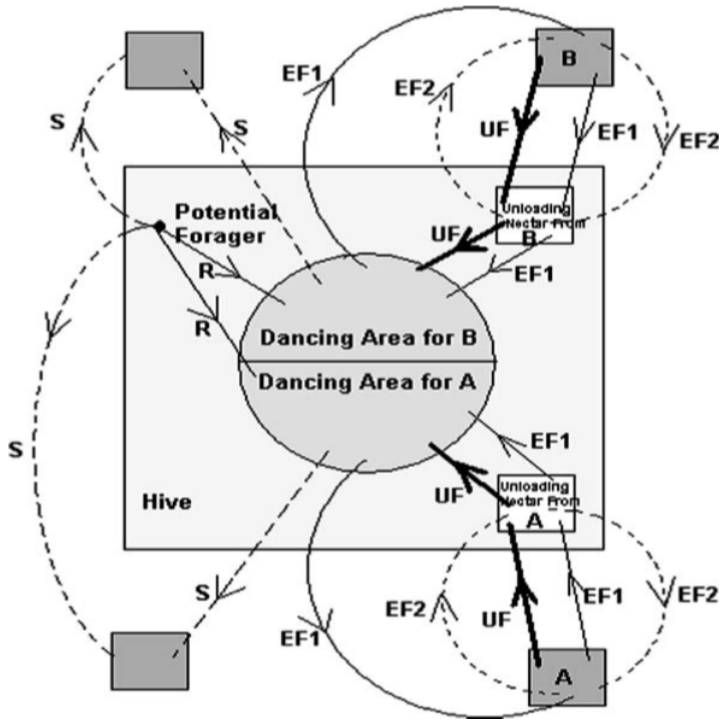
3. Unemployed foragers: called to a forager bee that searching for a site(food source) to exploit. Even a scout bee or an onlooker bee can be an unemployed bee. The average number of scout bees is about 5 to 10 percent.

One of the most important issues that shape collective knowledge of swarm is exchanging information between bees. Rather than exchanging information among bees the most important part of beehive is dancing area. The exchange of information or communications of bees about food sources quality happens in dancing area. The related dance is called waggle dance. Onlooker bee can watch many daces on the dance floor that they contain information about current resources and choose a most profitable site or food source to employ herself. Probability of choosing more profitable food sources is great because most of information circulating aroud is about or related to most profitable food sources. Hence, the recruitment is proportional to profitability of a food source [87].

For better understanding basic behavior of forager bees let us explain through Figure 2.18 that is obtained from [86].

A and B are two found food sources. At start point, a potential forager bee, starts as unemployed forager bee. As it mentioned before forager bees have no knowledge about food sources at this time, so there are two possible way to this kind of bee:

1. It can start searching around the nest spontaneously as a scout for food sources (S on Fig. 1.18).
2. Based on watching the waggle dances, it can be a recruit and search for a food source(R on Fig. 1.18).



**Figure 2.18** A graphical sample for behavior of bees based on [80].

The bee memorizes the location of food source after a food source has found and it will begin to exploiting the food source. At this stage, the bee becomes an employed forager bee. It takes possible amount of nectar to hive and unload it to food store. After unloading nectar into food store there are three possibilities for a forager bee:

1. Dancing and recruiting bees before returning to the same food source (EF1).
2. If food source is abandoned it could be an uncommitted follower (UF).
3. Searching a food source without recruiting bees (EF2).

All bees start foraging simultaneously. The experiments confirmed that new bees begin foraging at a rate proportional to the difference between the eventual total number of bees and the number of bees presently foraging [80].

### **2.3.2. Artificial bee colony (ABC) algorithm**

There are three types of bees in the artificial bee colony: employed bees, onlookers and scouts. Onlooker bees waiting on dancing area due to choosing a food source, Employed bees going to the food source visited by itself previously, Scout bees search randomly for food sources.

In artificial bee colony algorithm, population or colony divides into two parts. Half of colony includes employed artificial bees and another half includes onlooker bees. In ABC algorithm there is only one employed bee for each food source. The employed bee becomes scout when her food source is strained.

General process of the ABC algorithm according to [81] is as below:

1. Initialize positions of food sources
2. Evaluate the quality (fitness) of food sources
3. Produce new solutions (food source positions) for the employed bees
4. Chose the food sources based on greedy selection
5. Compute of probability and fitness
6. Produce new food sources for onlookers
7. Select food sources based on greedy selection
8. For abandoned food sources, change their employee bees roles as a scout to explore new food sources
9. Memorize the best food source of the cycle
10. Repeat steps 3-9 until to reach maximum number of cycles

At the initialization level, ABC algorithm selects a set of food sources by the bees randomly then their nectar amount must be examined. After that, bees must share information about food sources by going to beehive and on the dancing floor with the other bees that waiting for that information.

After initialization and sharing information every employed bee goes to the food source area that visited by herself before at the previous cycle and then try to choose a new food source via visual information in the neighborhood of previously selected area.

At the other stage, onlooker bees preferences to select food sources depend on the nectar information that share by employed bees on the dancing area. So probability of choosing a food source by onlooker increases when nectar amount of a food source increase, in the other word there is a direct relationship between probability of choosing a food source by an onlooker bee and information about nectar amount of that food source. A new food source could be determined randomly by scout bee when the food source is abandoned.

Positions of food sources denote solutions and nectar amount of them indicate fitness in the ABC algorithm. Also sum of employee and onlooker bees is equal to number of solutions or population size.

First of all, the ABC generates initial population randomly. Population size denote by SN and  $x_i$  ( $i = 1, 2, \dots, SN$ ) represent food source positions, that it is multidimensional vector. D represents number of dimensions in a vector.  $C = 1, 2, \dots, C_{\max}$ , represent number of cycles of ABC algorithm. Real bees select a new food source based on comparison of visual information that they gain from area but in ABC model, bees do not use any information to make comparison and they select a food source position randomly as it described in (22). After finding a new food source with higher amount of nectar or higher fitness it memorize just a new food source forgets old one. Otherwise the position of previous one is kept. After this, they share their information with other onlooker bees.

Choosing a food source via an onlooker bee depending on  $p_i$  that it calculated from expression (21):

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \quad (21)$$

$fit_i$  is the fitness value of position  $i$ .

SN is number of food sources or population size and it is equal to number of employed bees (BN).

Expression (22) is used to produce a new food position from old one:

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \quad (22)$$

where  $k \in \{1, 2, \dots, BN\}$  and  $j \in \{1, 2, \dots, D\}$

$k$  and  $j$  chosen randomly.

$\phi_{ij}$  is a random number in the range of  $[-1, 1]$ .

Margin between  $x_{ij}$  and  $x_{kj}$  has decreasing trend according to expression (22) also disorder on the position  $x_{ij}$  too.

If a parameter which is produced with this operation exceeds upper or lower limits it set to limit values. Abandoned food sources will be replaced by new ones by randomly produced positions. If a food source does not have improvement in specific number of cycles or iterations it will be abandoned. After position  $v_{ij}$  produced, its fitness value will be evaluated and if it is better than old one, it will be replaced.

Four different selection processes of ABC is as follow:

- (1) Searching area by artificial onlooker bees as mentioned in (21) forms a global selection process.
- (2) Local selection could be done by artificial employee bees and onlooker bees according to expression (22).
- (3) Local selection process is based on a greedy algorithm if new position is better than previous one it will kept but otherwise previous will kept.
- (4) Scout bees do a random selection.

### **2.3.3. Related works with ABC**

Since the ABC algorithm proposed by Karaboga [75], it drew attention of many researchers. It was applied to many optimization problems successfully. In this part, we will give brief review of previous works such as comparison of ABC algorithm with other well-known algorithms, ABC for feature selection and ABC for parameter optimization especially for SVM or SVR.

Karaboga and Basturk [77] applied ABC, Genetic Algorithm (GA), Particle Swarm Algorithm (PSO) and Particle Swarm Inspired Evolutionary Algorithm (PS-EA) for optimizing multi-dimensional numerical functions and compared results that produced by them. The results showed that the ABC algorithm has better performance than others.

GA, PSO and PS-EA are chosen because they are in the category of swarm intelligence and population based algorithms like ABC. Five benchmark functions that are chosen for this study are as follow:

1. Griewank; this function presents interdependence among the variables. Some techniques optimize every variable independently and because of overcome to failure of such techniques Griewank is used.
2. Rastrigin; this is a multimodal function based on Sphere function but with cosine modulation to produce many local minima. Also the minima are regularly distributed. Other significant characteristic of this function is that optimization algorithm can be trapped in a local optimum solution.
3. Rosenbrock; in this function variables are strongly dependent and it is used to test of performance of many optimization algorithms.
4. Ackley; there are many local minima in his function. Algorithms with wider search are could be able to pass local optimum and achieve better results.
5. Schwefel; In this function there are too many peaks and valleys and second best minimum is far from global minimum, so many algorithms can be trapped in a local minimum.

All mentioned algorithms tested with 500, 750 and 1000 iterations, also for 10, 20 and 30 dimensions. Population size for all of them is 125.

The results of this paper shows that only in Schwefel function for dimension 20 and 30, PS-EA and GA have better performance than ABC but with increasing iteration size ABC has a good performance on these two. According to [77], the ABC algorithm is quite successful in optimizing multivariable and multimodal functions and also has the ability to get out of a local minimum.

There is another study [76], about performance of the ABC. It compared ABC performance with differential evolution (DE), particle swarm optimization (PSO) and evolutionary algorithm (EA) for several multi-dimensional numeric problems. This paper used some test functions with different population sizes (10, 50, and 100) and different cycles to analysis ABC under different control parameter. The results show that the ABC has better performance than mentioned algorithms too.

Another similar study in comparison of the ABC with other algorithms for different test functions and parameters is done by [86]. It approved the previous results and show that the performance of the ABC is better than others with less control parameters.

#### **2.3.4. Feature selection based on ABC**

Feature selection is one of most important and critical concepts in data mining especially in preprocessing stage. Feature selection has direct impact on performance and accuracy of data mining algorithms and the computational complexity due to redundant, irrelevant and noisy features in the dataset. Feature selection extracts the relevant subset of features or variables from data set. There are a lot of studies in this field with different algorithms.

There are many studies about parameter optimization by ABC but there are a few studies for feature selection that we are going to discuss two of them here. Firstly, SyarifahAdilah et al.[35] proposed a feature selection method based on the ABC for using with SVM classifier for Hepatocellular carcinoma (HCC) data. In the other word it applied Artificial Bee Colony (ABC) algorithm to perform feature selection on mass spectrometry data.

Hepatocellular carcinoma (HCC) is a type of liver cancer which most commonly occurs among African and Asian people. This kind of cancer frequently happens among men than women and usually caused by scarring of the liver called as cirrhosis. The goal is to distinguish between HCC and cirrhosis cases [35].

Results (subset of features) that provided by the ABC algorithm, used for classification of HCC data with Support Vector Machine (SVM) classifier. Also 4-fold cross validation incorporates with SVM as validation method. Numbers of employed bees was set to 50 and number of features is 6. The number of onlooker bees was set two times of the numbers of employed bees. Also maximum cycle number (MCN) was set to 50. The results based on the ABC algorithm was compared with feature selection based on ant colony optimization algorithm (ACO). The following Pseudo Code shows the process of the feature selection based on the ABC exactly copied from [35]:

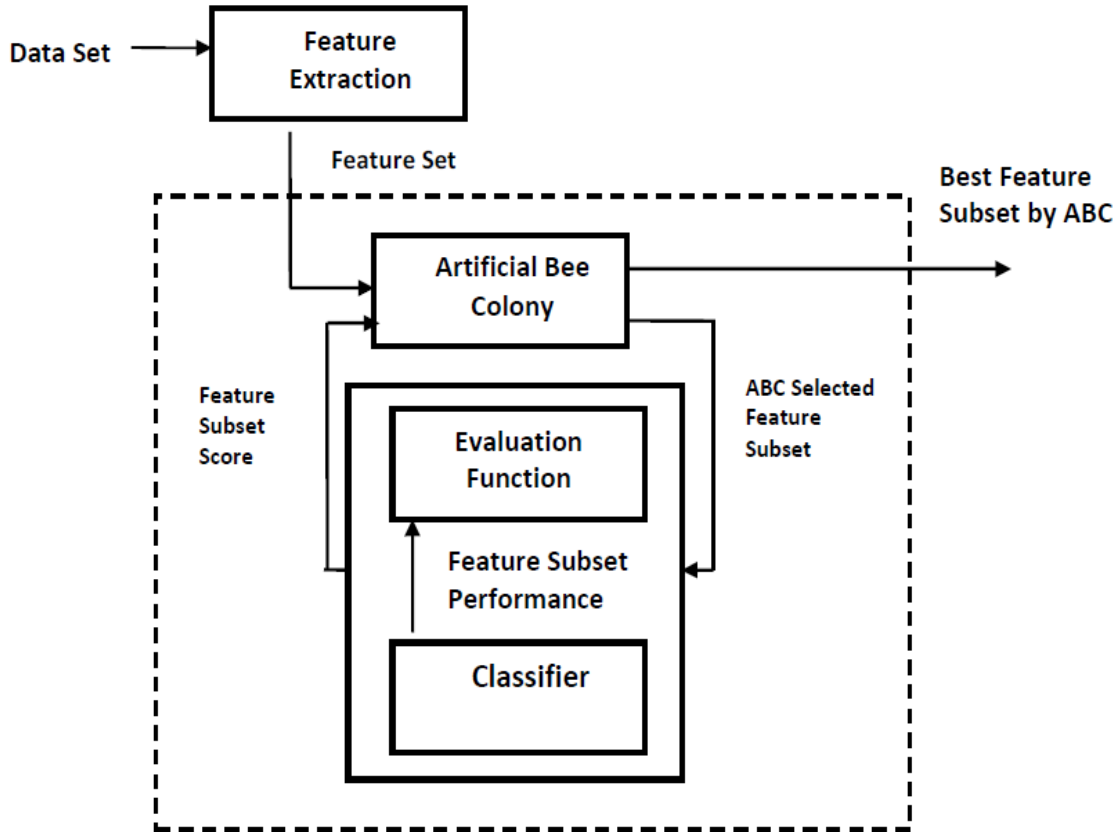
*1. Initialise population of solution  $X_{i,d}$  and limit trail<sub>i</sub>,*

for  $i=1,2,\dots,SN$  ;  $d= 1,2,\dots,D$ .

2. Evaluate population
3. Cycle=1
4. Repeat
5.     for  $i=1$  to  $SN$  do
6.             Produce new food source  $V_i$  for the current employed bee  $X_i$  using and evaluate the nectar quality.
7.             Apply greedy selection between  $V_i$  and  $X_i$  and select the better one.
8.             If solution  $X_i$  doesn't improve,  $trail_i=trail_i +1$ , Otherwise  $trail_i= 0$ ;
9.     End for
10.    Calculate the probability  $P_i$  for every solution.
11.     $t=0, i=1$
12.    Repeat
13.         If  $random < P_i$  then
14.             Produce a new food source  $V_i$  for onlooker bee and evaluate the nectar quality.
15.             Apply greedy selection process between  $V_i$  and  $X_i$  then select the better one.
16.             If solution  $X_i$  doesn't improve,  $trail_i=trail_i+1$ , Otherwise  $trail_i= 0$ ;
17.              $t=t+1$
18.         end if
19.    until ( $t=2SN$ )
20.    if  $\max(trail_i) > limit$  then
21.         Replace  $X_i$  with new randomly produced solution
22.    end if
23.    Memorize the best solution achieved
24.    cycle=cycle+1
25. until (cycle= MCN)

Results show that the ABC outperform ACO but both of them achieve 100 percent sensitivity and specificity.

Schiezaro and Pedrini [88] proposed another feature selection model based on the ABC, the results used for C4.5 classification algorithm and ten data sets that obtained by UCI [71] for evaluating results. Figure 2.19 represents the block diagram of feature selection algorithm based on the ABC.



**Figure 2.19** Block diagram of ABC based feature selection in [88].

Accuracy of each possible solution in the ABC calculated by applying 10-fold cross validation. A binary bit string assigned to each bee and length of it represent number of features. If a bit is 1 it means that that feature is selected else if it is 0, it means that it is not selected. Population size was set to  $2 * \text{No. of features in the data set}$ . Dimension of the population is number of features in dataset. Number of runs was set to 10 and Maximum cycle number was set to number of features. Steps of the proposed algorithm exactly according to [88] is as following:

1.  $\text{Cycle} = 1$
2. Initialize ABC parameters
3. Evaluate the fitness of each individual feature
4. Repeat
5. Construct solutions by the employed bees
  - i. Assign feature subset configurations (binary bit string) to each employed bee
  - ii. Produce new feature subsets  $v_i$

- iii. *Pass the produced feature subset to the classifier*
  - iv. *Evaluate the fitness ( $fit_i$ ) of the feature subset*
  - v. *Calculate the probability  $p_i$  of feature subset solution*
6. *Construct solutions by the onlookers*
- i. *Select a feature based on the probability  $p_i$*
  - ii. *Compute  $v_i$  using  $x_i$  and  $x_j$*
  - iii. *Apply greedy selection between  $v_i$  and  $x_i$*
7. *Determine the scout bee and the abandoned solution*
8. *Calculate the best feature subset of the cycle*
9. *Memorize the best optimal feature subset*
10. *Cycle = Cycle + 1*
11. *Until pre-determined number of cycles is reached*
12. *Employ the same searching procedure of bees to generate the optimal feature subset configurations*

Finally results have compared with ACO based feature selection. For most of data sets ABC based method proved better performance, Also ABC increased classification accuracies up to 12% compared to others.

### **2.3.5. SVM parameter optimization based on ABC**

There are many studies about parameter optimization for Support vector machines because selecting appropriate hyper parameters of SVM has great impact on performance and accuracy of it. In the case the ABC there several studies have done and most of them try to overcome parameter selection problem by modified versions of the ABC, because of weak exploitation of basic version of the ABC. With good exploitation techniques results could be good enough.

Mustaffa and Yusof [89] proposed a hybrid model of enhanced artificial bee and least square support vector machines for price forecasting. In mentioned study [89], for Least Squares Support Vector Machines (LSSVM), regularization parameter,  $\gamma$  and kernel parameter,  $\sigma^2$  were involved in optimization process through enhanced ABC algorithm. The enhancement, refer to modification that cause better exploitation for bees. The results of enhanced artificial bee colony-least squares support vector machine were compared to standard ABC-LSSVM and Back Propagation Neural Network. For mentioned study, four energy fuels price time series are used. All of time series are in daily basis, from December 1997 to November 2002. There are 15 features and they

normalized before training or testing. The metric for evaluation of fitness is Mean Absolute Percentage Error (MAPE). In order to improve exploitation, Levy Mutation and Simple Mutation are implemented. Figure 2.20 represents the process of proposed method by [89].

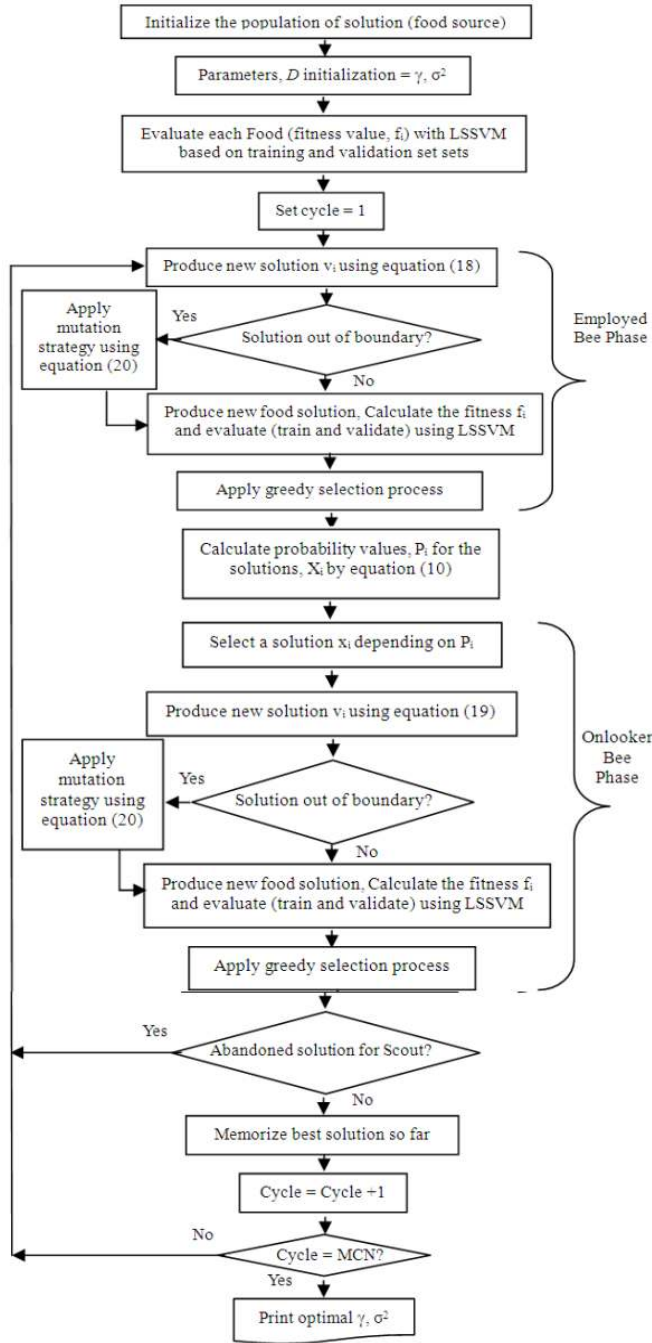


Figure 2.20 process of of eABC-LSSVM in [89]

By the way both simple mutation and levy mutation are applied in both employee bee and onlooker bee phases.

The ABC parameters were set as follow:

- i. Size of employed or onlooker bees (SN) was 10
- ii. Limit was set to  $SN * D$
- iii. Maximum Iteration was 100

The Artificial neural network parameters were set as follow:

- i. Hidden Node was 15.0
- ii. Learning rate was 0.9
- iii. Momentum was 0.7

According to the results in [89], the ABC perform better than ANN and as well as enhanced ABC is performed better than standard ABC.

Another problem that need optimization is real and reactive power tracing in a deregulated power system. Mohd Herwan et al.[92] proposed a hybrid method of the ABC and least squares support vector machine (LS-SVM) to solve this matter. The ABC algorithm is used to provide an optimum regularization

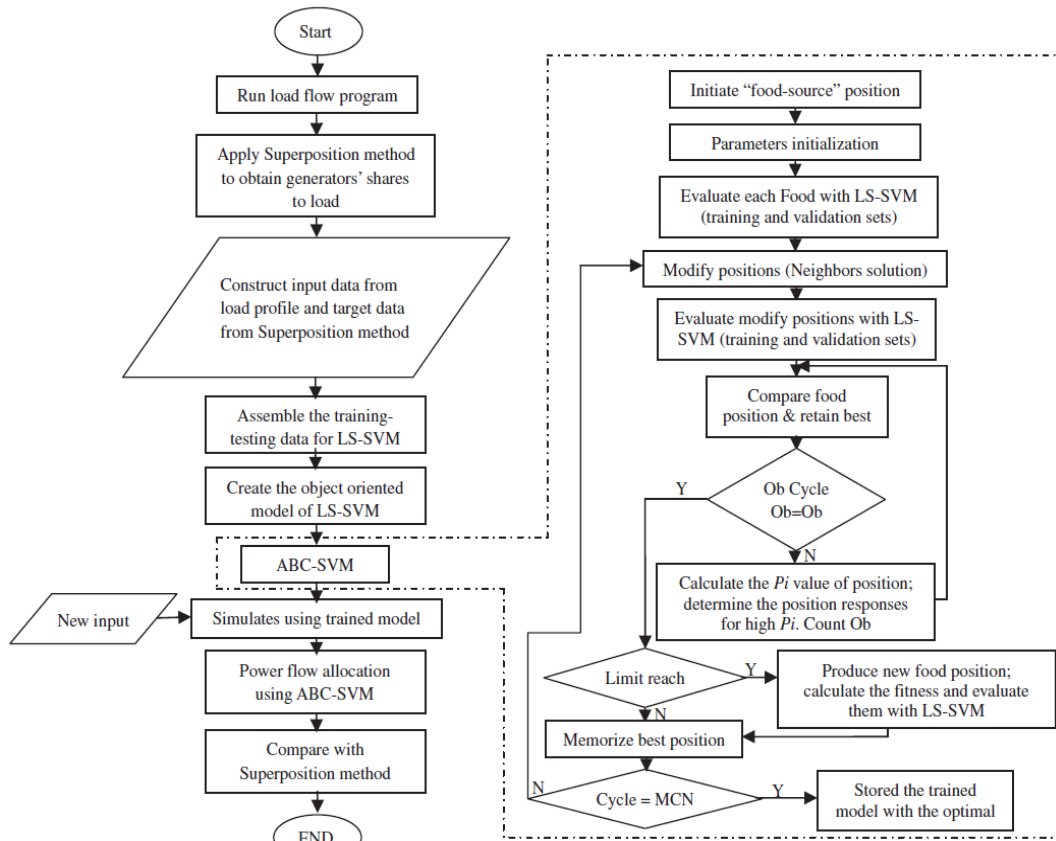
Parameters for LS-SVM (gamma, Kernel RBF parameter and sigma2). Based on power flow solution and power tracing procedure by Superposition method, the description of input–output for training and testing data are created. The generators' contributions to real and reactive loads in the test system are expected can be traced accurately by proposed ABC-SVM model [92]. Also this paper used Cross validation as validation method. In this paper standard Matlab LS-SVM toolbox is used [93]. input samples for training is provided by daily load curve and performing load flow analysis for every hour of load demand using the MATPOWER software package [94]. Training data is provided by the tracing results of Superposition method. The ABC algorithm parameter setting is as follows:

- i. Number of colony size or NP = 20.
- ii. The number of food sources = NP/2.
- iii. Limit = 2.

iv. Maximum cycle,  $MCN = 30$ .

This result for parameters of the ABC is based on trial and error method and it is chosen after trying some other values and finally the optimal values are as mentioned. Fitness value is calculated in term of MSE. In this paper 48 samples used for training and 72 samples used for testing. Flowchart of the proposed method that contains main steps of it, is exactly shown in Figure 2.21.

In addition, results of the proposed method Compared with genetic algorithms and it showed very good performance against GA. Moreover, with respect to amount of training samples this method showed very good performance too.



**Figure 2.21** Flow of proposed method in [92]

## 2.4. $\nu$ -SVR

In recent years, Support Vector Machines (SVM) attracted more attention for their outstanding performance. SVM inspired by VC dimensional theory and statistical learning theory [20] and also it can solve both classification and regression problems. In comparison with other machine learning techniques such as ANN, SVM has some considerable advantages as below:

- i. SVM solves a quadratic programming problem and make sure when algorithm provides an optimal solution, it is unique.
- ii. SVM derives a sparse and robust solution by maximizing the margin between the two classes [22].
- iii. SVM is based on the structural risk minimization principle.
- iv. Different kernel functions can be use due to solve linear and nonlinear problems.
- v. The kernel function can be useful to prevail the “curse of dimension”.

In the training phase of SVM there are two goals: minimization of training error and complexity of the model. Due to solve regression problems such as electricity market price prediction, Support Vector Regression (SVR) can be suitable based on mentioned advantages over other approaches mentioned before. SVR is a version of the SVM used for solving regression problems. In this study, SVR preferred because of mentioned characteristics of data and advantages over time series and ANN models.

SVR has been applied to many fields such as optimal control, image segmentation and etc. In SVR, there are a set of hyper-parameters must be chosen such as kernel parameter  $\gamma$ , the regularization constant  $C$  and epsilon  $\epsilon$ . Schölkopf et al in [21] proposed selecting  $\nu \in (0, 1)$  instead of parameter  $\epsilon$  because of difficulty with selection of  $\epsilon$  that is called  $\nu$ -SVR.  $\nu$ -SVR simplifies this issue by making  $\epsilon$  as part of optimization problem [21]. The new parameter ( $\nu$ ) can control the number of support vectors and training errors efficiently. Also  $\nu$ -SVM enables us avoid  $\epsilon$  (accuracy parameter) in regression problems and  $C$  (regularization constant) in classification cases [21], [95]. The theoretical interpretation is that  $\nu$  is an upper bound on the fraction of margin errors, although  $\nu$ -SVR produces a more generalized regression than other machine learning methods such as artificial neural networks [22]. In the following,

theory of v-SVR will be discussed briefly and detailed proofs are in [21] and [95].

For a given set of data points  $\{(x_1, y_1), \dots, (x_N, y_N)\}$ ,  $X_i \in \mathbb{R}^n$  is input vector and  $y_i \in \mathbb{R}^1$  target values, the primal problem of v-SVR is as (23):

$$\underset{w, b, \varepsilon, \xi_i, \xi_i^*}{\text{minimize}} \frac{1}{2} \|W\|^2 + C \cdot \left( v\varepsilon + \frac{1}{N} \sum_{i=1}^N (\xi_i + \xi_i^*) \right) \quad (23)$$

Subject to

$$(\langle w \cdot x_{-i} \rangle + b) - y_{-i} \leq \varepsilon + \xi_{-i}$$

$$y_i - (\langle w \cdot x_i \rangle + b) \leq \varepsilon + \xi_i^*$$

$$\varepsilon \geq 0 \text{ And } \xi_i, \xi_i^* \geq 0 \text{ for } i=1, \dots, N.$$

$0 \leq v \leq 1$ ,  $C$  is the regularization parameter,  $\varepsilon$  is allowed error for data point  $X_i$ ,  $\xi_i^*$ ,  $\xi_i$  are slack variables,  $N$  is the number of data points. There are two goals here. The first is to minimize the training error and the second is to find a solution with minimum error. By using the Lagrange multiplier the following dual optimization problem shows up in (24) according to [21]:

$$\underset{\alpha_i, \alpha_i^*}{\text{maximize}} \frac{-1}{2} \sum_{i=1}^N \sum_{j=1}^N (\alpha_i - \alpha_i^*) (\alpha_j - \alpha_j^*) \langle x_i \cdot x_j \rangle + \sum_{i=1}^N (\alpha_i - \alpha_i^*) y_i \quad (24)$$

Subject to

$$\sum_{i=1}^N (\alpha_i - \alpha_i^*) y_i$$

$$\sum_{i=1}^N (\alpha_i + \alpha_i^*) \leq C \cdot v$$

$$\alpha_i, \alpha_i^* \in \left[ 0, \frac{C}{N} \right]$$

$\alpha_i$  and  $\alpha_i^*$  are the Lagrangian multipliers. In this way nonlinearity could be achieved by mapping  $x$  into a high dimension space  $f$  through a map function  $\phi$  and the purpose is to find  $f(x) = \langle w \cdot \phi(x) \rangle + b$ .

In the case of kernel function, choosing the radial based function (RBF) seems reasonable most of times. It can handle cases with nonlinear relationship between class label and attributes. In other words, it maps samples into higher dimension space nonlinearly. Also RBF has less numerical difficulties in comparison with polynomial kernel and it has less hyper parameters than polynomial kernel. The following expression  $K(X_i, X_j) = \exp\left(-\gamma \|X_i - X_j\|^2\right)$  represents the RBF kernel and  $\gamma$  is the margin of RBF kernel.

## **Chapter 3: MATERIAL AND METHOD**

### **3.1. Data Extraction**

#### **3.1.1. Data**

In recent years, electricity supplier companies changed from monopoly utilities to deregulated competitive electricity market. Electricity price prediction is a critical issue to risk management strategies, decision making and strategy development process for market contributors in deregulated electricity market. Electricity prices have some particular characteristics such as non-stationarity, non-linearity and time-varying volatile structure. Despite many studies to predict of electricity market price, it is still needed more accurate price forecasting or prediction methods. In order to achieve better results, feature selection is one of the most important issues that must be considered to build a model [96, 97]. Electricity price has the following characteristics:

- i. non-linear
- ii. time variant
- iii. non-stationary
- iv. volatile signal with multiple periodicity
- v. high frequency components
- vi. significant outliers

There are some complications in dealing with electricity market price. It is impossible to store electricity. In electricity market, there are hourly, daily and seasonal ambiguities. Seasonal uncertainty could be rapid changes in weather, fuel price and behavior of market contributors. The mentioned reasons make electricity prices much more difficult.

The data used in this paper, provided from Turkish Electricity Market Reconciliation Center (<https://www.pmum.gov.tr>). Electricity prices from 15/04/2013 to 15/05/2013 are used as training and testing data. The data between 15/12/2012 and 15/05/2013 are used for correlation that is shown in Table 2.1. Electricity prices have some specific characteristics such as volatility, non-linearity and time-varying. Most studies are based on time series analysis and they also used lagged price (LP) and load values of past hours as features. So potentially, there are many features in this line of study.

Most of studies are based on time series analysis and also used lagged price and load

values of past hours as features. So potentially there could be many features. The following features retrieved from [96, 97]:

$P_{h-1}, P_{h-2}, P_{h-3}, P_{h-12}, P_{h-23}, P_{h-24}, P_{h-25}, P_{h-36}, P_{h-47}, P_{h-48}, P_{h-49}, P_{h-60}, P_{h-71}, P_{h-72}, P_{h-73}, P_{h-95}, P_{h-96}, P_{h-97}, P_{h-119}, P_{h-120}, P_{h-121}, P_{h-143}, P_{h-144}, P_{h-145}, P_{h-167}, P_{h-168}$ .

In this study, we add other features such as moving average based on past prices to improve prediction results. A moving average (MA) is a lagging indicator based on past prices and also it is commonly used in financial applications. Three types of moving averages used in this study: simple moving average (SMA), exponential moving average (EMA) and weighted moving average (WMA).

SMA is simple average of a defined number of time periods which is calculated from following formula:

$$SMA = \frac{P_M + P_{M-1} + \dots + P_{M-(n-1)}}{n} \quad (25)$$

Which  $P_M, P_{M-1}, \dots, P_{M-(n-1)}$  are same hours in the previous days.

WMA gives different weights to data at different positions which is calculated from following formula:

$$WMA = \frac{nP_M + (n-1)P_{M-1} + \dots + 2P_{M-(n+2)} + P_{M-(n+1)}}{n + (n-1) + \dots + 2 + 1} \quad (26)$$

Which  $P_M, P_{M-1}, \dots, P_{M-(n-1)}$  are same hours in the previous days.

EMA gives bigger weight to recent prices which is calculated from following formula:

$$EMA = \frac{P_1 + (1-\alpha)P_2 + (1-\alpha)^2P_3 + (1-\alpha)^3P_4 + \dots}{1 + (1-\alpha) + (1-\alpha)^2 + (1-\alpha)^3 + \dots} \quad (27)$$

Which  $P_1, P_2, \dots, P_M$  are same hours in the previous days.

### 3.1.2. Correlation

Correlation refers to statistical relationship between variables or dependency between them. There are some correlation coefficients for measuring degree of correlation mostly represented with  $\rho$  or  $r$ .

One of the most common measurements for degree of correlation is Pearson correlation coefficient that is only used for showing linear relationship for two variables but

sometimes may be it exists for when one of variables is a nonlinear function of other. Other correlation coefficients have been developed to be stronger than Pearson correlation coefficient in dealing with nonlinear relationships. In other words, they are sensitive to nonlinear relationships.

Pearson product-moment correlation coefficient or briefly Pearson's correlation is one of the most common measurements for correlation. It is calculated by dividing the covariance of the two variables by product of their standard deviation.

The population correlation coefficient  $\rho_{X,Y}$  between two random variables  $X$  and  $Y$  with expected values  $\mu_X$  and  $\mu_Y$  and standard deviations  $\sigma_X$  and  $\sigma_Y$  is defined as:

$$\rho_{X,Y} = \text{corr}(X, Y) = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y} \quad (27)$$

Where  $E$  is the expected value operator,  $\text{cov}$  means covariance, and,  $\text{corr}$  a widely used alternative notation for Pearson's correlation [100].

For calculating Pearson's correlation both of standard deviations must be finite and nonzero variables. The correlation value cannot exceed +1 or -1 and also correlation coefficient is symmetric  $\text{corr}(X, Y) = \text{corr}(Y, X)$ . In the best condition when there is a positive or increasing linear relationship value of correlation is +1. Also in the best condition when there is a negative or decreasing linear relationship value of correlation is -1. 0 denote no relationship between two variables. Closer to +1 and -1 values determine strong relationship.

Spearman's rank correlation coefficient or Spearman's rho, mostly marked by denoted by the Greek letter  $\rho$  (rho) or as  $r_s$ . It is a nonparametric measure of dependency between two variables. It uses a monotonic function to illustrate dependency between variables. Perfect Spearman correlation (+1 or -1) occurs when each variable is a perfect monotone function of the other and there is no repeated data value. Spearman's coefficient is suitable for continuous and discrete variables. Following formula computes Spearman's coefficient.

$$\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}} \quad (28)$$

$X_i$  and  $Y_i$  are converted ranks to  $x_i$  and  $y_i$ .

Because of mentioned characteristics of electricity price data, in this study, Spearman's rank correlation coefficient is used to analyze features before using them in training. Table 3.1 demonstrates Spearman's correlation coefficient results with respect to target price for features. In other words LPs of different times and MAs are correlated with each other in the period from 15/12/2012 to 15/05/2013. Eg. LP-24 means the correlation of the price of a specific hour (target price) with the price of 24 hours ago. Only the values above 0.6 are taken.

**Table 3.1** Spearman's correlation coefficient results for features

Feature Name	Correlation Coefficient	Sig.	Feature Name	Correlation Coefficient	Sig.
LP-24	.822	.000	SMA-5	.798	.000
LP-25	.747	.000	SMA-6	.800	.000
LP-48	.761	.000	SMA-7	.817	.000
LP-49	.692	.000	EMA-2	.830	.000
LP-72	.731	.000	EMA-3	.833	.000
LP-73	.664	.000	EMA-4	.832	.000
LP-96	.709	.000	EMA-5	.831	.000
LP-97	.644	.000	EMA-6	.830	.000
LP-120	.696	.000	EMA-7	.834	.000
LP-121	.632	.000	WMA-2	.829	.000
LP-144	.698	.000	WMA-3	.830	.000
LP-145	.633	.000	WMA-4	.829	.000
LP-168	.761	.000	WMA-5	.827	.000
SMA-2	.799	.000	WMA-6	.826	.000
SMA-3	.801	.000	WMA-7	.828	.000
SMA-4	.799	.000	Hour	.422	.000

In the same fashion, SMA-2 depicts the correlation of the price of a specific hour (target price) and the simple moving average of the price of the same hour in the preceding two days and EMA-7 shows the correlation of the price of a specific hour (target price) and the exponential moving average of the price of the same hour in the preceding seven days.

Each type of moving average with 2 to 7 periods is added to the feature set. Correlation coefficient analysis shows that moving averages have higher correlation than raw lagged

prices.

Finally, HOUR represents hour in day between 1 and 24, it is not highly relevant but it is significant.

### 3.2. Fitness Definition

In this study, k-fold cross validation method is used. Cross-validation reflects the performance of regression and it is necessary to use in training and testing of SVR models. In this method all dataset is divided into k subsets. For k times, one of k subsets is used as testing set and the remaining ones are used as training set. Performance of a particle or a food source is measured by mean square error (MSE). Then, the average errors of all k trials are computed. One of the advantages of this method is that it is not important how we divide dataset and every data point could be in the testing set just once, and k-1 times in training set. For greater k, there is a reduction in variance. The disadvantage of this method is that we need to rerun the training and testing for k times for evaluation. In other words it is a time consuming process.

MSE is calculated as follows:

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - d_i)^2 \quad (29)$$

Where y is real value and d is predicted value.

### 3.3. Simultaneous Parameter Determination And Feature Selection Based On PSO

In this study, the constricted version of the PSO proposed in [45] used to simultaneous parameter determination and feature selection for SVR. As it mentioned before, modified velocity equation used and velocity clamping technique is not needed anymore. Equation (4) represents the modified velocity equation. In order to compute constriction factor, equation (5) and equation (6) used respectively. Based on studies about inertia weight in [43, 44], a decaying inertia weight used and it was calculated via equation (7). At first, due to better global search for the algorithm and later to better local search. The pseudo code of the basic PSO is as follows with respect to [98]:

*For each particle*

*Initialize particle*

*END*

*Do*

*For each particle*

*Calculate fitness value*

*If the fitness value is better than the best fitness value (pbest) in history*

*set current value as the new pbest*

*End*

*Choose the particle with the best fitness value of all the particles as the gbest*

*For each particle*

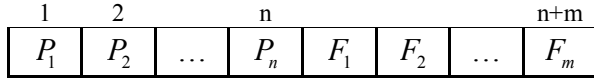
*Calculate particle velocity according equation (4)*

*Update particle position according equation (2)*

*End*

*While maximum iterations or minimum error criteria is not attained*

At the first step or in the initialization step, dimensions for each position start with random numbers and velocities initialized with random numbers too. Particles' velocities on each dimension are limited to maximum velocity value represented by  $V_{\max}$ . Also, v-SVR used with RBF kernel used to compute fitness. Figure 2.1 illustrates the solution representation by each particle.



**Figure 3.1** Representation of a particle in proposed PSO

$P_i$  : Represent SVR kernel parameters needed to be optimized.

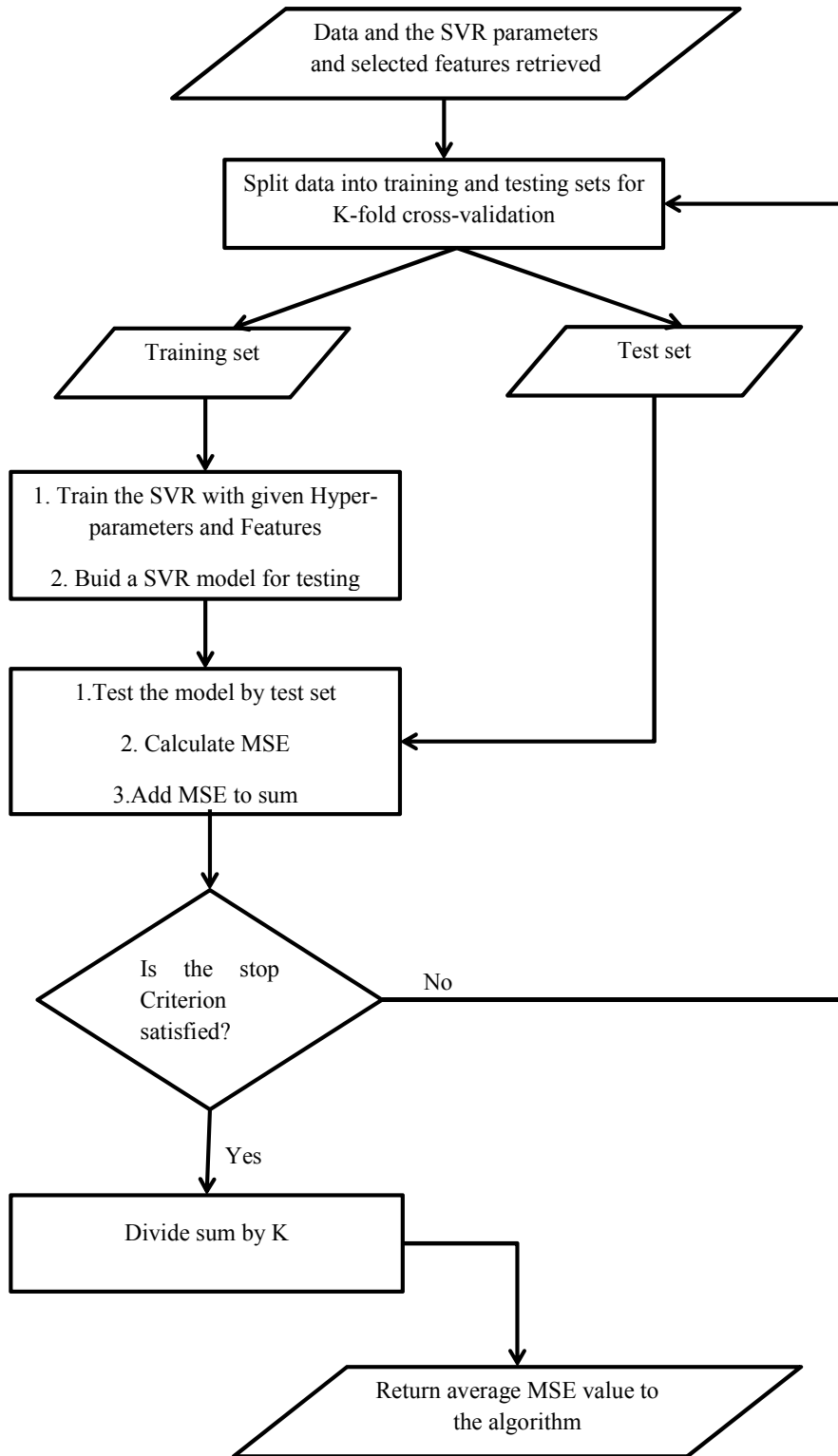
$F_i$  : Represent features needed to be selected.

n: is number of parameters

m: is number of features

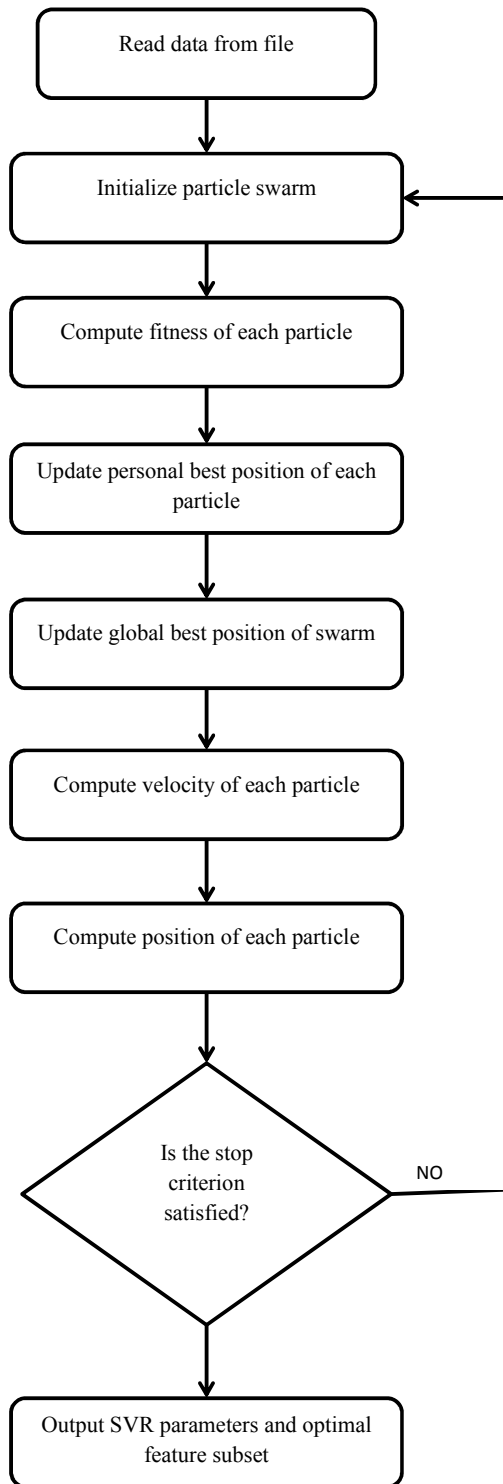
So, there are n+m dimensions for each particle. All of dimensions are continuous variables. Every dimension constrained to maximum and minimum values. The SVR parameters have their own boundaries but all features have same boundaries.

In this study we used k-fold cross validation method. Cross-validation reflects the performance of regression and it is necessary to use in training and testing of SVR models. In this method all dataset divide into k subsets. For k times, one of k subsets used as testing set and other remaining used as training set. After all, the average error of all k trials computed. One of the advantages of this way is that it is not important how we divide dataset and ever data point could be in testing set just once, and k-1 times in training set. For greater k, there is a reduction in variance. Finally disadvantage of this way is that we need to rerun training and testing for k times for evaluation. In the other word it is time consuming process. In this study, cross-validation used to compute fitness for the SVR. Flowchart of computing fitness is as Figure 3.2:



**Figure 3.2** Flowchart of computing fitness

Figure 3.3 shows the procedure of the PSO used in this study:



**Figure 3.3** shows the procedure of the PSO

### **3.4. IPSO**

The main idea of this implementation came from study in [37] (Improved binary particle swarm optimization (IBPSO)). In fact, IPSO is continuous version of the IBPSO for simultaneous feature selection and parameter optimization of  $v$ -SVR that proposed in this study. The reason of proposing this method is to overcome trapping in a local optimal. In the standard PSO, if the global best particle trapped in a local optimum, search area of the entire swarm restrict to that search area. Therefore, the algorithm could not be able to provide appropriate results. In proposed method by [37], global best particle reinitialize when after some iterations it remains unchanged. Value of the global best must be considered before updating positions of the swarm, if global best remains unchanged for predefined number of iteration, it means that it may stick in a local optimum. Chuang et al.[37], set this value to three. In other words, if global best remains unchanged after three iterations, the global best position is reset to zero but particle best value remains as it is. After changing global best to zero, particles that trapped in a local optimum will regulate their positions toward the global best. This method is based on standard PSO but there is just an additional part for reinitialize global best particle position, so general process and fitness evaluation are same as standard PSO.

### **3.5. MPSO**

The basic idea of this part, came from study in [42] (Modified binary particle swarm optimization (MBPSO)) which is a mixture of reset best particle of swarm mention before and local search operator that is similar to mutation. MPSO is continuous version of the MBPSO for simultaneous feature selection and parameter optimization of  $v$ -SVR that proposed in this study. Results of experiments show that IPSO method does not act well in problem addressed in this study, so due to avoid trapping in local optimal there is needed another approach that could be able to pass local optimal with higher probability. Another approach was proposed by Lee et al. [74] is to allow small mistakes (mutations) in positions of particles. After updating position of each particle according to equations (4) and (2), Mutation applied to each dimension of particle with probability of  $r$ . Value of  $r$  compute by one divide by number of dimensions, which is ensure that at least one of dimensions can be mutated. In this study, we assign a random value between boundaries of each dimension to that dimension. The mutation should cause

diversity in the swarm without too much randomization of population. This method is combination of two ideas: one is to reinitialize global best particle's position when it does not improve after predefined number of iterations, second is to apply mutation to particles after updating them in standard version of PSO.

### **3.6. The ABC Method For Simultaneous Parameter Determination And Feature Selection**

The ABC algorithm discussed in in session 1.3. In this part of study the ABC will reviewed briefly. Then structure of food source and the procedure of the algorithm explained.

Each point in search space of the ABC corresponds to a food source (individual solution) that can be exploited by a bee. Amount of nectar of a food source indicate fitness value of that solution. There are three types of bees in the artificial bee colony: employed bees, onlookers and scouts. Employed bees exploit the food source and give information about the quality of food source to onlooker bees. Onlooker bees wait on dancing area for choosing a food source. Scout bees search randomly for food sources. If a food source contains more nectar, onlooker bees choose it with a higher probability. The employee bee can become a scout bee when the food source that it worked on is abandoned. This is controlled by a parameter called "limit." In ABC algorithm, population or colony is divided into two parts. Half of the colony is comprised of employed artificial bees and the other half contains the onlooker bees [77]. There is only one employed bee for each food source which means that number of food sources is equal to the number of employee bees. The goal of whole colony is to provide maximum amount of nectar.

After a short description of the ABC above, in the next parts of study three main phases of the ABC will be explained separately.

#### **3.6.1. The employed bee phase**

In this phase, employee bees update their food sources via following equation:

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \quad (30)$$

Where  $v_{ij}$  and  $x_{ij}$  represent the new and old solution (food source) in  $j^{\text{th}}$  dimension of the

$i_{th}$  individual,  $k \in \{1, 2, \dots, BN\}$  and  $j \in \{1, 2, \dots, D\}$ ,  $\phi_{ij}$  is a random number in the range of  $[-1, 1]$  and  $k$  is chosen randomly.

If a parameter which is produced with this operation exceeds upper or lower limits, it set to the limit values. After the new food source is updated, the correspondent fitness value of the food source is evaluated. In order to remember the food source afterwards, greedy selection algorithm is used. In other words, after finding a new food source with higher amount of nectar or higher fitness, it memorizes the new food source and forgets the old one. Otherwise the previous one is reserved.

### 3.6.2. The onlooker bee phase

After completing search for food sources in the employee bee phase, the employee bees share their information about food sources with onlooker bees. Onlooker bees choose a food source based on the amount of nectar (fitness value) of that food source. Probability of choosing a food source is calculated as follows:

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \quad (31)$$

$fit_i$  is the fitness value of position  $i$  correspondent to nectar amount of  $i_{th}$  food source. SN is number of food sources or population size.

### 3.6.3. Scout bees phase

If one or more food sources do not show improvement in the predefined cycle called ‘‘limit,’’ it will be replaced according to following equation:

$$x_{id} = x_d^{min} + r(x_d^{max} - x_d^{min}) \quad (32)$$

Where  $x_d^{min}$  and  $x_d^{max}$  represent the lower and upper boundary in dimension  $d$ ,  $r$  is a random number between  $[0, 1]$ . A formula was proposed for assigning a number for constant parameter limit before run and it is as follows: Limit = SN \* D [86].

Figure 3.4 represents the structure of food source in ABC.

1	2	...	n	n+1	n+2	...	n+m
$P_1$	$P_2$	...	$P_n$	$F_1$	$F_2$	...	$F_m$

**Figure 3.4** Representation of food source in ABC algorithm

$P_i$ : Represents SVR parameters needed to be optimized.

$F_i$ : Represents a feature to be selected.

$n$ : is number of parameters

$m$ : is number of features

$v$ -SVR parameters are placed first followed by the features in the food source. There are  $n+m$  dimensions for each food source. All of the dimensions are continuous variables. Every dimension is constrained to maximum and minimum values. The SVR parameters have their own boundaries, but all features have the same boundaries that are between  $[F_{\max}, F_{\min}]$  where  $F_{\max}$  is the upper bound of the feature and  $F_{\min}$  is the lower one. In this study, there are three parameters of  $v$ -SVR that must be determined. When the value of any dimension is greater than  $F_{\max}$  or less than  $F_{\min}$  it set to  $F_{\max}$  or  $F_{\min}$  accordingly. If value of dimension corresponding to a feature is greater or equal to  $(F_{\max} + F_{\min})/2$  it means that the correspondent feature is selected. Otherwise it is not selected. The algorithm of the proposed method is as follows:

1. *Read electricity prices data and initial the ABC parameters such as the maximum number of iterations, etc.*
2. *Initialize the population (food sources) including hyper parameters and features.*
3. *Calculate the fitness value for each member of population or each food source with  $v$ -SVR based on cross validation.*
4. *Set cycle=1.*
5. *Produce new solution  $v_i$  for the employed bees based on (30).*
6. *Calculate fitness value with  $v$ -SVR based on cross validation.*
7. *Select either the new or the old food source based on greedy selection.*
8. *Compute probability values  $p_i$  based on (31).*
9. *Produce the new solutions  $v_i$  for the onlooker bees through existing solutions  $x_i$  based on the  $p_i$  values.*
10. *Calculate fitness value with  $v$ -SVR based on cross validation.*
11. *Select either the new or the old food source based on greedy selection.*
12. *Locate abandoned solutions for the scout bees. If there is any abandoned solution, alter it with a new randomly produced solution using (32).*
13. *Memorize best solution found yet.*
14. *If number of iterations is equal to the maximum number of iterations given stop else go to step 5.*

## Chapter 4: DISCUSSION AND EVALUATION

In this part of thesis, at first, experiments of three different versions of the PSO for parameter and feature selection will be discussed separately, after that three proposed methods will be compared together. Furthermore outcome of ABC and comparison with PSO based methods will be discussed.

### 4.1. Experiments Of The Standard Version Of The PSO

There are two purposes behind these experiments: First is to find a better parameter set for the PSO based on Table 2.1 and second is to figure out performance of the proposed model. In order to do this experiment, three different parameter settings for the PSO mentioned in Table 2.1, used with different iteration numbers and also swarm sizes. Most of studies used lower number of swarm sizes such as 30 and 40, in this study, 20, 30, 40, 50, 60 and 80 swarm sizes used to test models. As it mentioned in literature review, maximum iteration number is 100 or 200 in most of papers. Due to inappropriate performance of model in 100 and 200 iteration number, experiments continued up to 500 and 1000. In other words, experiments carried on against 100, 200, 500 and 1000 maximum iteration number. v-SVR parameters needed to be determined are  $v$ ,  $C$  and  $\gamma$ .  $v$  is in the range of  $[0, 1]$  and range of both  $C$  and  $\gamma$  are between 1000 and -1000. Value of  $K$  in  $K$ -fold cross validation is 5.

Table 4.1 shows experimented results with the following PSO parameters:

- i.  $C_1=C_2=1.4962$
- ii. Inertia weight is 0.7968

Each number of swarm size (20, 30, 40, 50, 60, 80) tested against 100, 200, 500, 1000 maximum iteration numbers. Value of  $K$  in  $K$ -fold cross validation is 5.  $v$  is in the range of  $[0, 1]$  and range of both  $C$  and  $\gamma$  are between 1000 and -1000. Number of features are 32. Total number of dimensions for each particle is  $32+3=35$ .

**Table 4.1** Results of PSO with parameters based on [47]

Max Iterations	Swarm Size	Average of 10 runs	Minimum of 10 runs
1000	20	484.47	174.07
	30	460.63	173.98
	40	457.29	160.73
	50	433.74	59.809
	60	383.67	55.806
	80	263.01	55.806
500	20	457.52	55.806
	30	373.69	95.383
	40	443.31	160.85
	50	371.58	55.806
	60	311.04	55.806
	80	424.12	174.04
200	20	485.9	180.07
	30	459.29	55.806
	40	456.5	55.806
	50	405.66	55.806
	60	474.93	293.77
	80	402.7	60.75
100	20	543.7	373.84
	30	538.61	373.67
	40	515.79	279.28
	50	533.67	307.83
	60	501.96	388.01
	80	489.93	371.6

Table 4.2 shows experimented results with the following PSO parameters:

- i.  $C_1=C_2=2$
- ii. Inertia weight is 1

Each number of swarm size (20, 30, 40, 50, 60, 80) tested against 100, 200, 500, 1000 maximum iteration numbers. Value of K in K-fold cross validation is 5.  $v$  is in the range of [0, 1] and range of both C and  $\gamma$  are between 1000 and -1000. Number of features are 32. Total number of dimensions for each particle is  $32+3=35$ .

**Table 4.2** Results of PSO with parameters based on [44]

Max Iterations	Swarm Size	Average of 10 runs	Minimum of 10 runs
1000	20	416.63	95.383
	30	499.18	174.07
	40	406.95	55.806
	50	435.38	160.85
	60	435.68	92.322
	80	349.68	55.806
500	20	412.96	55.806
	30	464.88	55.806
	40	450.37	240.05
	50	361.55	55.806
	60	475.87	234.33
	80	408.49	60.72
200	20	459.47	96.68
	30	455.55	55.806
	40	487.57	342.17
	50	415.92	95.383
	60	441.61	174.03
	80	417.59	109.02
100	20	551.09	268.78
	30	536.04	279.09
	40	502.06	270.1
	50	514.81	334.35
	60	480.66	178.04
	80	490.61	184.46

Table 4.3 shows experimented results with the following PSO parameters:

- i.  $C_1=C_2=2$
- ii. Inertia weight linearly decrease from 0.9 to 0.4 according to equation (7)

Each number of swarm size (20, 30, 40, 50, 60, 80) tested against 100, 200, 500, 1000 maximum iteration numbers. Value of K in K-fold cross validation is 5.  $v$  is in the range of [0, 1] and range of both C and  $\gamma$  are between 1000 and -1000. Number of features are 32. Total number of dimensions for each particle is 32+3=35.

**Table 4.3** Results of PSO with parameters based on [44]

Max Iterations	Swarm Size	Average of 10 runs	Minimum of 10 runs
1000	20	427.38	95.383
	30	402.23	174.04
	40	395.75	205.85
	50	401.23	174.04
	60	365.39	92.322
	80	261.4	55.806
500	20	402.44	55.806
	30	375.93	55.806
	40	393.68	59.809
	50	378.93	55.806
	60	337.99	201.28
	80	348.16	55.806
200	20	470.01	174.68
	30	480.55	238.35
	40	431.89	55.806
	50	401.86	240.05
	60	398.46	160.85
	80	397.09	55.806
100	20	558.81	377.915
	30	531.95	297.81
	40	505.23	377.03
	50	520.65	317.03
	60	490.48	344.26
	80	491.69	315.08

In order to find better parameter settings for the PSO, results in each Table 4.1, 4.2 Table and Table 4.3 compared with other two. In iteration size 100, average of 10 runs in Table 4.3 is better than Table 4.1 for swarm sizes 30, 40, 50 and 60. In iteration size 200, average of 10 runs in Table 4.3 is better than Table 4.1 for swarm sizes 20, 40, 50, 60 and 80. In iteration size 500, just half of results are better, but in iteration size 1000 all results in Table 4.3 are better than Table 4.1. Generally, it could be inferred from results that results in Table 4.3 are better than Table 4.1.

In iteration size 100, average of 10 runs in Table 4.3 is worse than Table 4.2 for all swarm sizes but they are too close to each other and the difference is not much. In iteration size 200, average of 10 runs in Table 4.3 is better than Table 4.2 for swarm sizes 40, 50, 60 and 80. In iteration sizes 500 and 1000, except two cases with little difference,

other cases in Table 4.3 are better than Table 4.2 and in some case with more difference. So based on mention reasons,  $C_1=C_2=2$  and inertia weight linearly decrease from 0.9 to 0.4 parameter setting chosen for the other experiments.

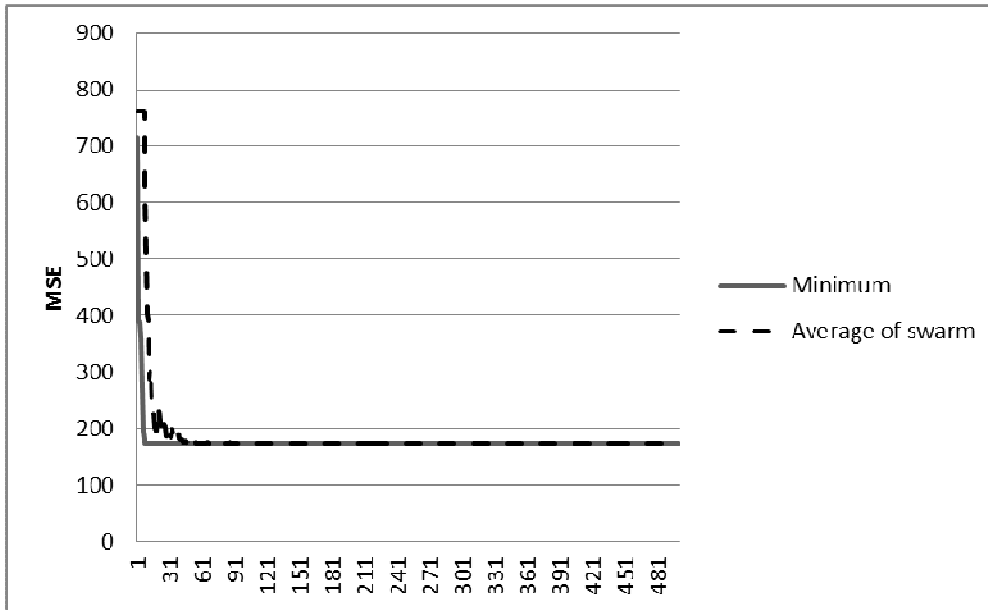
There are some anomalies in the results due to tendency of the PSO to a premature convergence. In other words, the PSO has a potential to trap in local optima. When this happens, with the most probability it stick there, other iterations become useless and there is no improvement in the results. Because of that, sometimes outcome of higher swarm size is not better than lower swarm size with the same number of iterations. Table 4.4 represents outcome of 10 runs with  $C_1=C_2=1.4962$  and  $w= 0.7968$ , Maximum iteration size of 1000 and swarm size of 80. In this table, every 100 iteration global best value of the swarm was recorded until the maximum iteration number. In iteration 1000, there are MSE numbers between 55.806 and 511.37. With respect to the outcome, it can be seen that in four of runs, algorithm stick in local optima far from an optimal solution could be achieved. Best found solution is with MSE of 55.806 and it shows that the algorithm could find better solutions but most of times it trapped in local optimal and could not pass them.

**Table 4.4** Actual MSEs of 10 runs of PSO

Run	Iteration										
	1	100	200	300	400	500	600	700	800	900	1000
1	741.23	256.34	199.75	174.07	174.04	174.04	174.04	174.04	174.04	174.04	174.01
2	579.01	572.49	570.48	106.07	92.322	92.316	92.316	92.316	92.316	92.316	92.316
3	616.3	143.71	117.86	117.86	117.86	117.86	117.86	117.86	117.86	117.86	117.86
4	687.84	193.07	174.07	174.02	174.02	174.01	173.96	173.95	173.95	173.95	173.93
5	702.7	568.66	524.61	503.12	497.72	497.46	496.87	495.92	495.92	495.9	493.83
6	710.78	582.11	570.17	510.98	507.02	506.99	506.83	506.31	506.31	506.3	506.11
7	671.87	219.11	59.809	55.806	55.806	55.806	55.806	55.806	55.806	55.806	55.806
8	626.87	574.62	574.62	549.38	542.16	501.84	498.87	497.99	494.45	490.23	490.23
9	594.13	151.22	123.33	117.86	117.86	117.86	117.86	117.86	117.86	117.86	117.86
10	670.98	588.99	560.8	519.06	516.85	511.42	511.42	511.38	511.38	511.37	511.37

Figure 4.1 shows a sample for trapping in local optima. In this sample, global best position of swarm stick in local optima and other particles adjust themselves according to global best. Grey line represents minimum solution found by the algorithm since the algorithm start and dash line shows average fitness of particles in every iteration. It could be seen that, in early iterations, a local optimal solution found by the algorithm and all

particles moved toward global best position. In the words, the algorithm converged to suboptimal solution. After iteration 85 all other iterations produced same results.



**Figure 4.1** trapping in a local optimal

In the standard version the PSO, most of times, the algorithm converge to a solution till iteration 400 and there is no considerable improvement in results after that.

## 4.2. Experiments Of The IPSO

Table 4.5 shows experiment results with the following PSO parameters:

- i.  $C_1=C_2=2$
- ii. Inertia weight linearly decrease from 0.9 to 0.4 according to equation (7)

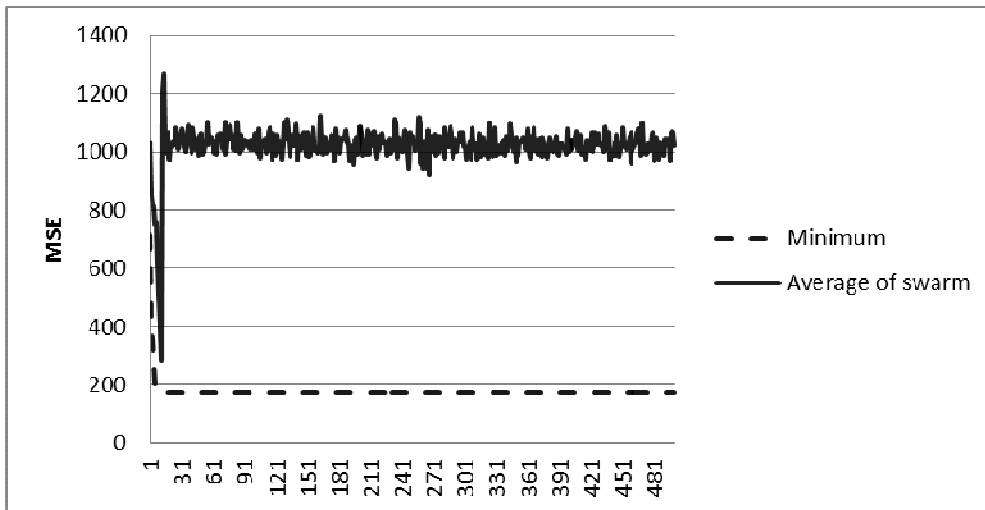
Each number of swarm size (20, 30, 40, 50, 60, 80) tested against 100, 200, 500, 1000 maximum iteration numbers. Value of K in K-fold cross validation is 5.  $v$  is in the range of [0, 1] and range of both C and  $\gamma$  are between 1000 and -1000. Number of features are 32. Total number of dimensions for each particle is 32+3=35.

**Table 4.5** Results of IPSO

Max Iterations	Swarm Size	Average of 10 runs	Minimum of 10 runs
1000	20	420.74	95.383
	30	385.1	174.04
	40	401.7	171.15
	50	403.82	92.322
	60	381.58	89.85
	80	258.293	55.806
500	20	402.42	55.806
	30	374.52	55.806
	40	448.37	55.809
	50	360.08	95.383
	60	340.29	162.28
	80	330.2	59.809
200	20	465.01	181.46
	30	489.98	220.68
	40	450.237	92.875
	50	388.652	150.509
	60	391.758	167.75
	80	387.96	55.806
100	20	522.87	207.17
	30	510.56	117.86
	40	513.64	377.03
	50	547.21	503.12
	60	481.41	257.85
	80	496.69	269.96

In iteration size 100, average of 10 runs in Table 4.5 are better than Table 4.3 for swarm sizes 20, 30, 40 and 60, also other two are worth. In iteration size 200, average of 10 runs in Table 4.5 are better than Table 4.3 for swarm sizes 20, 40, 50 and 80, also other two are worth. In iteration size 500, average of 10 runs in Table 4.5 are better than Table 4.3 for swarm sizes 20, 50 and 80, also other three are worth but with less difference. In iteration size 1000, average of 10 runs in Table 4.5 are better than Table 4.3 for all swarm sizes. Because of mentioned reasons, it could be said that the IPSO, generally produced better results than standard version of the PSO. Most of times results are better but there is no significant difference between them. According to references [24, 30], this strategy, does not operate well on higher dimension data likewise in experiments of [24, 30], there is 2-2.5% improvement in performance of model for different datasets.

According to reference [30], this strategy cannot act well in higher dimensions and could be trapped in a local optimal as well. The experiments in this study support the mentioned idea specially when there are too many local optimal. Figure 4.2 shows a sample for trapping in local optima for continuous version of the IPSO. Dash line represents minimum solution found by the algorithm since the algorithm start and Black line shows average fitness of particles in every iteration. Figure 4.2, represents how this algorithm could be trapped in local optimal. At early iterations, global best particle of the swarm stick in a local optimal and particles converge toward global best. But after three iterations that global best value stay unchanged, the position of the global best particle reset. Sudden increases illustrate reset of global best particle. Unfortunately, resetting global best is not enough in this case and local optimal cannot be passed. Other issue that must be considered here is that algorithm is continue to searching other areas around despite Figure 4.1 that all particles adjust themselves based on global best until they all are same value or very close positions together. In other words, if speed of convergence toward global best is fast enough even resetting global best is not enough to avoid trapping in a local optimal. In conclusion this strategy can increase chance of finding better solutions but in the case of this study, that chance is not enough to achieve better performance.



**Figure 4.2** trapping in a local optimal

### 4.3. Experiments Of The MPSO

Table 4.6 shows experiment results with the following PSO parameters:

- i.  $c_1=c_2=2$
- ii. Inertia weight linearly decrease from 0.9 to 0.4 according to equation (7)

Each number of swarm size (20, 30, 40, 50, 60, 80) tested against 100, 200, 500, 1000 maximum iteration numbers. Value of K in K-fold cross validation is 5.  $v$  is in the range of [0, 1] and range of both C and  $\gamma$  are between 1000 and -1000. Number of features are 32. Total number of dimensions for each particle is  $32+3=35$ .

**Table 4.6** Results of MPSO

Max Iterations	Swarm Size	Average of 10 runs	Minimum of 10 runs
1000	20	378.81	55.806
	30	395.47	55.806
	40	339.37	55.806
	50	374.54	55.806
	60	360.63	117.86
	80	371.7	55.806
500	20	421.82	83.69
	30	350.73	241.71
	40	385.5	73.489
	50	347.13	55.806
	60	350.12	55.806
	80	296.83	95.383
200	20	483.95	75.96
	30	480.65	175.4
	40	420.7	94.12
	50	375.17	69,7
	60	380.6	55.806
	80	383.85	183.7
100	20	521.3	207.17
	30	539.85	466.67
	40	481.16	229.77
	50	505.42	501.96
	60	490.49	496.31
	80	465.1	173.29

In iteration size 100, average of 10 runs in Table 4.6 are better than Table 4.5 for swarm sizes 20, 40, 50 and 80, also other two are worth. In iteration size 200, average of 10 runs in Table 4.6 are better than Table 4.5 for swarm sizes 30, 40, 50, 60 and 80, also other one is worth. In iteration size 500, average of 10 runs in Table 4.6 are better than Table

4.5 for swarm sizes 30, 40 50 and 80, also other three are worth. In iteration size 1000, average of 10 runs in Table 4.6 are better than Table 4.5 for swarm sizes 20, 40 50 and 60, also other two are worth. Based on above, results of MPSO are better most of times. As well as in comparison with Table 4.3, results of modified PSO are better too. But this not absolute, because even with mutation and reset together it is obvious that in this case it does not meet expectations . MPSO experiments like in [42] results have 2-3% improvements in comparison with IPSO. But in this case even with these improvements outcome is not close to expectation.

#### **4.4. Comparison Of The PSO Based Methods**

Because of premature convergence of the PSO, improved PSO was used to avoid trapping in local optimal. Table 4.7 shows results of proposed methods for simultaneous parameter determination and feature selection based on PSO in one place to better comparison. There is improvement in IPSO results but it is not as expected as well as MPSO. Both of IPSO and MPSO are proposed to avoid premature convergence and/or increase chance of avoiding to trap in local optimal. Outcome of the MPSO shows that it has greater chance to find optimal solutions due to better performance in comparison with PSO and MPSO . But difference between best found solutions and average of 10 runs indicates that PSO based models are not suitable for solving this problem and they can not discover search space efficiently.

**Table 4.7** Comparison of proposed PSO based methods

Max Iterations	Swarm Size	PSO	IPSO	MPSO
1000	20	427.38	420.74	378.81
	30	402.23	385.1	395.47
	40	395.75	401.7	339.37
	50	401.23	403.82	374.54
	60	365.39	381.58	360.63
	80	261.4	258.293	371.7
500	20	402.44	402.42	421.82
	30	375.93	374.52	350.73
	40	393.68	448.37	385.5
	50	378.93	360.08	347.13
	60	337.99	340.29	350.12
	80	348.16	330.2	296.83
200	20	470.01	465.01	483.95
	30	480.55	489.98	480.65
	40	431.89	450.237	420.7
	50	401.86	388.652	375.17
	60	398.46	391.758	380.6
	80	397.09	387.96	383.85
100	20	558.81	522.87	521.3
	30	531.95	510.56	539.85
	40	505.23	513.64	481.16
	50	520.65	547.21	505.42
	60	490.48	481.41	490.49
	80	491.69	496.69	465.1

## 4.5. The ABC Experiments

### 4.5.1. Test of limit value

As it mentioned before there are not many parameters to set in ABC algorithm. There are three Parameters needed to be set before run which are Limit, Population size or number of bees and Max iteration size. When a food source which its nectar is abandoned by the bees, replace with a new source by scouts. In ABC, this process simulated by producing a random position and assigning new position with abandoned food source. Also, In ABC algorithm, if a position cannot be improved through a predetermined number of cycles, then that food source is granted to be abandoned. The value of predetermined number of cycles is a very important parameter of the ABC algorithm and it called "Limit". If value of limit is high then the algorithm could be stick in local optimum and also cannot discover the other positions in search space that may have the best solution.

Otherwise, if it is a low number, the algorithm cannot have better performance on local search, Thus, this value playing a very important role in the ABC and its value must be an optimized value and truly setting of this value cause a better performance of the ABC. Karaboga and Akay [86], proposed a formula for assigning a number for constant parameter limit before run and it is,  $\text{Limit} = \text{SN} * \text{D}$  where SN is equal to half of population size and D is number of dimensions. So as first experiments on ABC, we start with this parameter. The aim of this experiment is to find out a value of the limit according to our problem. In other words, we want to make sure that the mentioned formula is providing a good solution to our problem.

Number of attributes for this experiment is 35 (3 for v-SVR and 32 features). We used one moth data which is containing 720 records so total number of attributes for optimization process is 35. Population size of the ABC algorithm is set to 50. Every result which come from the ABC algorithm or every combination of parameters and features that validated through a 5 fold cross validation. According to mentioned formula that defines limit number, this number must be  $25*35= 875$ . We start with limit 200 and increased limit by 100 in each step. Then run the ABC algorithm 10 times for different number of iteration sizes and compute the averages of all minimum solutions found in that specific iteration size and placed them in Table 4.8.

According to Table 4.8, results around the suggested limit number are not always better than the others. In Table 4.8, cells with better MSE values than limit size 875 are filled by grey color. In 1500 and 1250 iteration sizes, results of 200, 300 and 700 limit value are far better than 875. Up to 200 iteration size results are not important because limit values strat from 200 and between 500 and 750 iteration sizes outcome is not clearly interpretable. Based on results most of times results based on proposed equation in [86] are better or close. So we accept the proposed method for defining limit number.

**Table 4.8** Analyzing ABC with different limit values

iteration limit	100	200	500	750	1000	1250	1500
200	523.48	484.49	365.9	333.98	292.3	196.96	178.54
300	521.91	504.09	308.01	240.89	190.19	181.25	181.23
400	536.88	503.26	359.17	299.61	296.72	274.04	262.87
500	521.64	490.35	344.69	332.78	330.03	291.15	252.06
600	526.84	484.88	352.38	300.28	272.16	221.13	168.57
700	525.33	487.67	305.48	192.56	180.86	180.8	180.79
800	514.21	481.66	394.83	355.92	337.04	317.57	314.06
875	512.87	474.1	345.23	314.95	255.92	197.34	185.3
1000	525.83	468.33	356.48	280.38	229.69	225.06	203.31

#### 4.5.2. Test of cross validation and dataset size

There are two opinions behind the next experiments; first, because results of most of studies about SVR are based on 5 fold cross validation and for greater k, there is a reduction in variance. But it is needed to rerun training and testing for k times for evaluation. In other word it is time consuming process. In this study, the aim is to acquire minimum MSE or highest model accuracy regardless of considering processing time. Therefore, due to acquire better accuracy and minimum error, model was tested with 5 and 10 fold cross validation. Second, most of studies about electricity market use one or two month data. Because each day of week has a different pattern of electricity consuming, for example Tuesday and Saturday are different, on Tuesday most of people spent most of time at work after a weekend, all of government building, companies and etc. are used but on the other side people do not use lots of energy at home. In opposite side on Saturday all or most of government places, banks, companies are closed and people spent more time at home and consuming energy pattern is different than Saturday. So we decided to examine each day's data differently due to get less error and most prediction precision in comparison with one month data.

For the first purpose, number of attributes for this experiment is 35 (3 for v-SVR and 32 features). One moth data is used, which is containing 720 records and total number of attributes for optimization process is 35. For each day we used data of that specific day of week for past 30 week, it make total number of records 720. In the other words, for each day of week, we separate that day's information from 30 week and combine them in the order of date as one dataset. Total number of features for this status is 32 and three

for v-SVR. Every result which come from the ABC algorithm or every combination of parameters and features that validated through a 5 fold cross validation. Whole of above settings are used with different population sizes (30, 40, 50). We run the ABC algorithm 10 times for different number of iteration sizes and compute the averages for all minimums found in that specific iteration size also save the minimums of 10 runs and placed them in Tables. Table 4.9 to Table 4.11 show the results for population sizes 30, 40, 50.

Next time all above mentioned settings and datasets used with 10 fold cross validation. We run the ABC algorithm 10 times for different number of iteration sizes and compute the averages for all minimums found in that specific iteration size also save the minimums of 10 runs and placed them in Tables. Table 4.12 to Table 4.14 show the results for population sizes 30, 40, 50.

Table 4.9 and Table 4.12 demonstrate results for population size 30 with 5 and 10 fold cross validation. Table 4.10 and Table 4.13 demonstrate results for population size 40 with 5 and 10 fold cross validation. Table 4.11 and Table 4.14 demonstrate results for population size 50 with 5 and 10 fold cross validation. Difference between results is obvious. Based on this results that shown in mentioned tables, there is a clear difference between using 5 fold cross validation with the same population size and 10 fold cross validation. Results based on 10 fold cross validation are much better than 5 fold most of times and we can achieve better performance of regression in the same setting. There are some anomalies ofcourse such as in Table 4.9 MSE of one month data for iteration 1500 is better than the same in Table 4.11 unlike expectation but majority of results show that 10 fold cross validation give better accuracy. But of course, it is more time consuming process than 5 fold.

For the second purpose of our experiment at first, we compare results of one month with different days in 5 fold cross validation. Each day of week that has better MSE than one month dataset filled with grey color. In 100, 200 and 500 iterations results based on different days are better than one month in average of 10 runs and even finding better minimums. But in 1000 and 1500 iterations it can not be determined with most certainty for 30 and 50 population sizes. Generally based on these observations (Table 4.9, Table 4.10, Table 4.11), it will be reasonable to divide dataset into seven different days and

continue to train model based on them. Because most of times MSE and minimums of different days are better than whole one month data in one place.

Secondly, we compare results of one month with different days in 10 fold cross validation. Table 4.12 to Table 4.14 show the results. The outcome is clearly shows that the results of one month are better than results based on days of week for 100 and 200 iterations for all population sizes. For 500, 1000 and 1500 iterations, training based on one month dataset absolutely give better results in the case of average of 10 runs and minimums found in 10 runs. Only in population size 30 it is inverse but in this case the algorithm found 28.813 MSE with average of 10 runs 202.19 that minimum found is far better than every minimum found by training each day of week. So based on mentioned reasons, one month data can be used to train model for electricity price forecasting and also it is less time consuming than divid data into seven days and train and test model base on them. In other case divid data into seven days could provide ability to forecasting a whole week with accuracy between 91.33% and 85.85% that some days have accuracy above 90% and others are worth.

**Table 4.9** Analyzing cross validation and separate dates effect with 30 population size and 5 fold cross validation

limit		100	200	500	1000	1500
D1	avr	462.68	396.34	257.81	166.28	148.29
	min	450.95	151.53	121.04	97.415	97.415
D2	avr	402.64	334.85	244.31	174	133.01
	min	367.6	245.43	142.05	131.16	131.16
D3	avr	357.68	342.68	271.61	225.19	224.92
	min	343.87	338.77	122.5	120.49	120.49
D4	avr	465.05	442.85	346.4	308.02	277.73
	min	445.79	428.9	158.92	158.92	158.92
D5	avr	456.88	414.55	361.68	286.06	283.54
	min	410.44	301.83	263.3	142.24	142.24
D6	avr	320.1	276	211.26	189.74	188.11
	min	315.82	159.25	92.083	92.083	92.083
D7	avr	295.43	282.86	209.57	164.09	154.6
	min	284.89	276.6	97.593	97.593	97.593
1 month	avr	529.15	502	350.56	221.16	179.56
	min	500.98	491.96	124.97	95.38	95.38

**Table 4.10** Analyzing cross validation and separate dates effect with 40 population size and 5 fold cross validation

limit		100	200	500	1000	1500
D1	avr	459.01	410.59	268.93	186.36	163.18
	min	446.9	289.24	121.04	97.416	97.416
D2	avr	417.85	350.09	253.23	162.76	164.67
	min	387.44	289.29	209.44	131.16	131.16
D3	avr	351.08	341.03	221.32	201.39	166.16
	min	343.17	337.33	137.61	136.2	136.2
D4	avr	455.88	418.49	322.8	253.42	228.78
	min	451.22	315.59	165.29	141.39	141.39
D5	avr	446.21	405.54	305.2	291.77	291.1
	min	428.6	285.84	151.62	151.62	151.62
D6	avr	316.79	309.69	231.77	220.52	218.74
	min	308.95	304.78	93.581	93.581	93.581
D7	avr	295.58	278.08	243.11	196.39	184.61
	min	289.51	248.68	182.67	97.593	97.593
1 month	avr	509.91	480.63	376.96	340.26	325.09
	min	496.43	426.03	238.36	167.69	160.86

**Table 4.11** Analyzing cross validation and separate dates effect with 50 population size and 5 fold cross validation

limit		100	200	500	1000	1500
D1	avr	460.52	357.76	210.9	97.419	97.417
	min	458.39	282.86	210.79	97.419	97.415
D2	avr	398.49	327.28	212.03	131.16	131.16
	min	292.95	254.63	199.76	131.16	131.16
D3	avr	354.51	340.08	221.92	219.15	235.43
	min	349.29	323.12	137.61	136.2	136.2
D4	avr	454.54	415.46	286.77	254.44	251.74
	min	446.15	339.67	173.02	158.92	141.39
D5	avr	450.31	406.41	323.16	267.24	265.92
	min	443.03	292.03	235.02	142.24	142.24
D6	avr	318.58	289.94	196.79	188.51	201.83
	min	310.17	232.91	92.083	92.083	92.083
D7	avr	294.58	284.1	208.33	194.54	194.52
	min	289.4	279.15	108.78	97.593	97.593
1 month	avr	516.53	450.97	315.36	224.02	205.17
	min	500.31	374.94	92.317	92.317	92.317

**Table 4.12** Analyzing cross-validation and separate dates effect with 30 population size and 10 fold cross validation

limit		100	200	500	1000	1500
D1	avr	468.78	354.92	189.75	80.46	80.46
	min	459.03	209.31	92.608	75.799	75.799
D2	avr	411.21	345.66	203.24	113	112.99
	min	384.22	278.45	103.06	103.06	103.05
D3	avr	351.2	340.12	217.1	135.1	135.1
	min	346.81	336.12	122.27	94.251	94.251
D4	avr	455.78	384.94	232.08	220.16	220.15
	min	440.22	203.16	125.06	122.57	122.57
D5	avr	453.05	428.51	342.76	272.58	272.51
	min	435.5	383.01	265.82	120.48	120.48
D6	avr	319.49	311.92	224.76	183.23	140.65
	min	316.03	306.09	118.8	78.706	78.706
D7	avr	308.95	227.15	180.26	173.3	173.18
	min	297.8	138.82	82.905	82.905	82.905
1 month	avr	496.05	476.16	324.65	209.23	202.19
	min	471.5	453.42	84.762	28.813	28.813

**Table 4.13** Analyzing cross validation and separate dates effect with 40 population size and 10 fold cross validation

limit		100	200	500	1000	1500
D1	avr	467.84	369.31	218.11	102.23	102.23
	min	456.47	277.07	76.718	76.718	76.717
D2	avr	426.2	372.54	186.88	104.17	103.63
	min	410.75	307.75	118.97	103.06	103.05
D3	avr	356.26	334.01	238.76	216.25	216.25
	min	343.38	309.02	119.53	119.53	119.53
D4	avr	448.2	398.88	252.12	230.7	161.99
	min	444.59	318.83	130.92	125.65	122.57
D5	avr	454.52	401.03	251.96	225.16	209.65
	min	437.99	340.27	135.26	107.88	107.88
D6	avr	320.91	307.45	233.11	189.71	184.86
	min	316.13	299.22	78.706	74.54	72.794
D7	avr	302.54	280.6	187.35	184.54	168.91
	min	290.36	261.68	88.82	82.905	82.905
1 month	avr	496	480.19	112.49	64.865	46.839
	min	491.13	468	53.395	28.813	28.813

**Table 4.14** Analyzing cross validation and separate dates effect with 50 population size and 10 fold cross validation

limit		100	200	500	1000	1500
D1	avr	463.68	372.29	198.41	84.663	75.798
	min	452.09	320.32	194.41	76.718	75.798
D2	avr	422.87	325.77	173.62	136.52	121.12
	min	416.28	253.31	103.06	103.06	103.05
D3	avr	346.88	313.82	180.5	121.33	121.33
	min	335.83	238.87	94.349	94.251	94.251
D4	avr	437.71	382.57	175.33	163.23	162.46
	min	400.29	288.67	125.06	125.06	122.57
D5	avr	423.22	361.07	302.27	254.32	218.77
	min	386.82	252.05	120.48	120.48	120.48
D6	avr	315.75	309.52	214.03	187.57	125.04
	min	312.97	305.89	121.15	72.794	72.794
D7	avr	297.75	280.31	124.45	88.517	88.517
	min	292.77	248.16	82.905	82.905	82.905
1 month	avr	519.13	454.42	137.02	64.865	64.865
	min	491.23	444.24	95.566	28.813	28.813

### 4.5.3. Testing parameters of ABC and comparison

There are three purposes behind the final experiments:

1. Find the best possible population size and max iteration number for the study problem.
2. Compare simultaneous feature selection with parameter optimization of v-SVR with all features.
3. Compare one and two month datasets performances.

For this experiment, we used two dataset, one month and two month data. In one month data there are 720 records and in two month data there are 1440 records. Number of attributes for this experiment is 35 (3 for v-SVR and 32 features) or dimension size is 35. The ABC algorithm run for different population sizes 10, 20, 30, 40, 50, 60, 80, and 100. Value of limit computed from following expression:  $Limit = SN * D$ . Every result which come from the ABC algorithm or every combination of parameters and features validated through a 10 fold cross validation. Then run the ABC algorithm 10 times for

different number of iteration sizes and compute the averages for all minimums found in that specific iteration size also save the minimums of 10 runs for those specific iterations and placed them in tables. For each population size, there two phases:

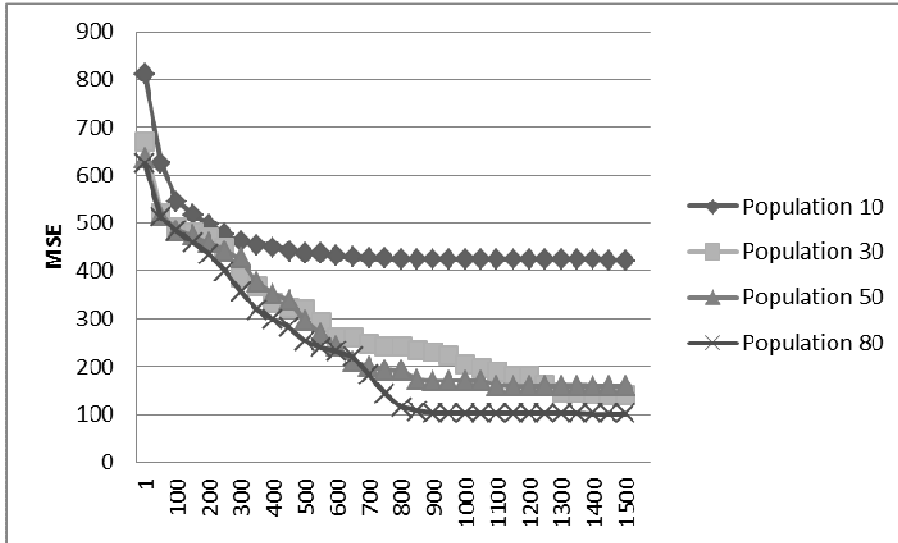
First; for testing algorithm just for parameter optimization of the SVR. It means that in every step, the ABC algorithm just try to optimize 3 parameters of v-SVR and all features sent to SVR as they are and there is no change in features number. In this way, we have three dimensions to optimize with fix number of features that is equal to 32.

Second, we test the ABC with simultaneous parameter optimization and feature selection. It means that in each step, we test different combination of features and parameters that produced by the ABC. In other words, in each trial, we send three parameter to SVR with different number of features for training and testing model. There are 35 dimensions needed to be optimized in this way.

Table 4.15 to Table 4.22 demonstrate results for 10, 20, 30, 40, 50, 60, 80, 100 population sizes with both one month and two months datasets with v-SVR parameter optimization and simultaneous parameter optimization and feature selection results for different iterations up to 1500 maximum iteration size.

First of all, with respect to Table 4.15 to Table 4.22 that demonstrate results for 10, 20, 30, 40, 50, 60, 80, 100 population sizes consequently. Each table contain both one month and two month datasets and they tested with 3 and 35 dimensions. For three dimensions (just parameter optimization of v-SVR), it could be seen that for all population sizes, the ABC optimize the results till 100 max iteration size and after that there is no obvious or considerable change in performance of algorithm. So based on the mentioned reason it can be inferenced that for just parameter optimization of v-SVR best possible outcome can be achieved with minimum population size (10) and maximum iteration size 100. In this way, there are only three dimensions and also all features accepted as they are. In this part the aim was only try to find best possible parameters for the v-SVR with all features that dataset contains. Of course, there are possible redundant features in the dataset that cause MSE. Experiments based on just parameter selection of v-SVR has done due to comparison with simultaneous parameter determination and feature selection to analyze efficiency of model, also efficiency of the model compared with PSO in the next parts of study.

Secondly, with respect to Table 4.15 to Table 4.22 that demonstrate results for 10, 20, 30, 40, 50, 60, 80, 100 population sizes that contain both one month and two months datasets, for 35 dimensions (3 for parameter optimization of v-SVR and 32 for features of one month and two months datasets), it can be inferred that for great number of dimensions, with increasing number of population size, the results get better or there is decreasing in average of MSEs for ten runs. Also, almost in all of the results for different iteration sizes outcome is getting improved considerably. At some point, there are anomalies for example in Table 4.21 and Table 4.19 that they hold results of population size 80 and 50 in maximum iteration sizes 1000 and 1500, average of MSE for ten runs are greater than the same for population size 50. Two reasons can cause this problem: first is the ABC algorithm can stick in a local optimum in some runs that cause to increasing average of minimum mean square error for ten runs. Second is because of nature of the ABC, this algorithm can not perform well in searching local positions for achieve an optimal solution that cause to increasing average of minimum MSE for ten runs too. But totally outcome getting better and in the greater iteration sizes it is much more obvious.



**Figure 4.3** effect of population size on results

Figure 4.3 represent effect of increasing population size on results in 1500 iterations for different population sizes. Because of avoiding complexity of graph just four population size used to analysis here. These are based on one month data. Based on Figure 4.3 with increasing population size curve falling faster and most of times in the same iterations

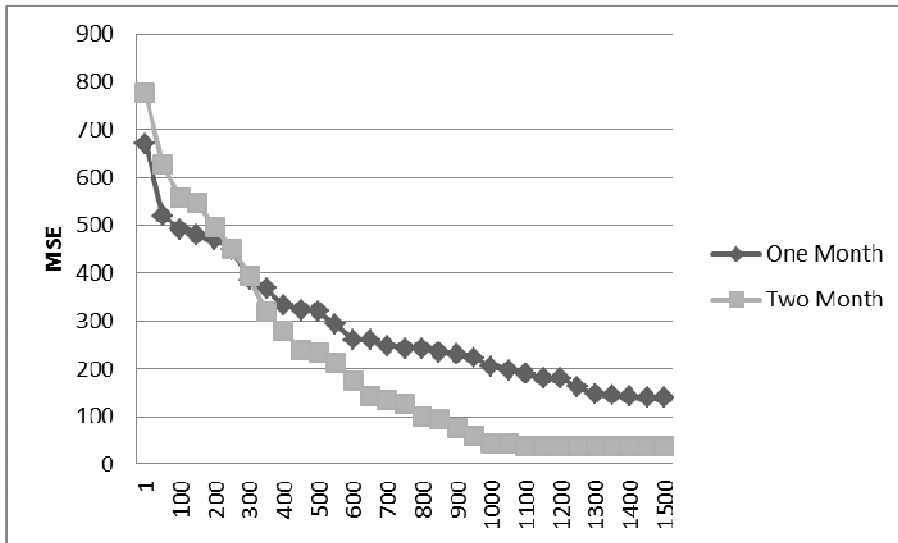
higher population has lower MSE. Higher population size has better chance to pass local optimal and find better solutions.

In this experiment with respect to Table 4.15 to Table 4.22 that demonstrate results for 10, 20, 30, 40, 50, 60, 80, 100 population sizes, it can be inferred that population size 60 hold good and acceptable results and averages of minimum MSE for ten runs are too close to best possible solutions found during different runs. In other words, averages of MSE for ten runs are approximately minimum between the other outcomes in above mentioned tables and also their values are close to found best possible MSE in different runs. So it can be inferred that under mentioned conditions for our purpose we can use setting of Table 4.20 to achieve better performance and less error rate in prediction of electricity market data.

For the second purpose of this experiment (Compare simultaneous feature selection with parameter optimization of v-SVR with all features), with respect to Table 4.15 to Table 4.22 that demonstrate results for 10, 20, 30, 40, 50, 60, 80, 100 population sizes, for all tables the outcomes of each dataset(one month and two months) for 3 dimensions (just parameter optimization of u-SVR) and 35 dimensions comparison shows that, simultaneous feature selection and parameter optimization has quite good performance than just parameter optimization in both lower and higher iteration sizes. In high iteration sizes the results are much better than lower iteration sizes; especially in high population sizes it is completely obvious. Thus based on mentioned results, simultaneous parameter optimization and feature selection method outperform just parameter optimization method in this case. Because rather than high correlation coefficient values for features some mostly relevant features can become irrelevant and some irrelevant become relevant when used together. Besides using some feature in dataset that can not provide useful information anymore can cause curse of dimensionality and decrease prediction accuracy.

For the third purpose of this experiment that is comparison of one and two month datasets performances, with respect to Table 4.15 to Table 4.22 that demonstrate results for 10, 20, 30, 40, 50, 60, 80, 100 population sizes, for all of table it is pretty obvious that results of two months dataset (average of minimum MSE for ten runs) are quite better than one month results. In this session, analysis of outcomes is based on simultaneous

parameter optimization and feature selection, because it is proven to be better than just parameter optimization. For lower population sizes, averages of minimum MSE are pretty more than best minimums that found in ten runs or best possible solution. Also in lower iterations averages of minimum MSE are quite more than best found minimum in ten runs too. But in higher iterations, averages of minimum MSE are close to found best solutions. In other words, distance between average of minimum MSE for ten runs and best possible solution with minimum of solutions found in ten runs is closer. It means that we can achieve best possible solutions in this area. But it does not mean that one month dataset could not possibly provide better solution because in Table 4.20, the best solution that found for one month dataset is 28.813 but for two months this is 37.597. In conclusion, performance of two months dataset is better than one month in even lower iterations but its processing time is much more than one month. Figure 4.4 shows MSEs of one and two month data till 1500 iterations . In mentioned figure two month data has steepy curve than one month and reached about 60 MSE till 1000 iterations.



**Figure 4.4** comparison of one and two month data

**Table 4.15** Population size 10

	NOD	AVR	100	200	500	1000	1500
1 MONTH	3	MIN	488.75	487.58	487.51	487.46	487.44
		AVR	487.47	487.43	487.43	487.4	487.4
2 MONTH	3	MIN	572.79	572.57	572.44	572.44	572.44
		AVR	572.44	572.44	572.44	572.44	572.44
1 MONTH	35	MIN	544.84	497.31	437.01	423.17	422.39
		AVR	500.44	447.26	382.18	382.18	382.18
2 MONTH	35	MIN	596.02	535.27	451.49	230.68	182.43
		AVR	548.64	502.16	85.895	44.165	38.044

**Table 4.16** Population size 20

	NOD	AVR	100	200	500	1000	1500
1 MONTH	3	MIN	487.65	487.59	487.42	487.41	487.4
		AVR	487.41	487.4	487.4	487.4	487.4
2 MONTH	3	MIN	572.48	572.44	572.44	572.44	572.44
		AVR	572.44	572.44	572.44	572.44	572.44
1 MONTH	35	MIN	535.6	431.49	317.92	277.77	255.67
		AVR	493.68	284.96	87.067	28.813	28.813
2 MONTH	35	MIN	568.41	430.7	107.77	58.327	61.311
		AVR	539.86	271.48	40.1	37.631	37.616

**Table 4.17** Population size 30

	NOD	AVR	100	200	500	1000	1500
1 MONTH	3	MIN	487.69	487.43	487.4	487.4	487.4
		AVR	487.43	487.4	487.4	487.4	487.4
2 MONTH	3	MIN	572.44	572.44	572.44	572.44	572.44
		AVR	572.44	572.44	572.44	572.44	572.44
1 MONTH	35	MIN	491.65	470.24	319.87	204.19	138.43
		AVR	478.29	435.58	129.29	28.813	28.813
2 MONTH	35	MIN	556.05	495.72	235.67	44.003	38.824
		AVR	533.77	366.94	64.28	37.657	37.657

**Table 4.18** Population size 40

	NOD	AVR	100	200	500	1000	1500
1 MONTH	3	MIN	487.59	487.44	487.4	487.4	487.4
		AVR	487.42	487.4	487.4	487.4	487.4
2 MONTH	3	MIN	572.44	572.44	572.44	572.44	572.44
		AVR	572.44	572.44	572.44	572.44	572.44
1 MONTH	35	MIN	500.69	469.68	298.52	179.28	162.29
		AVR	473.77	439.47	132.43	67.814	28.813
2 MONTH	35	MIN	558.16	375.59	73.203	50.621	49.431
		AVR	542.34	292.24	40.008	37.78	37.614

**Table 4.19** Population size 50

	NOD	AVR	100	200	500	1000	1500
1 MONTH	3	MIN	487.45	487.42	487.4	487.4	487.4
		AVR	487.42	487.4	487.4	487.4	487.4
2 MONTH	3	MIN	572.44	572.44	572.44	572.44	572.44
		AVR	572.44	572.44	572.44	572.44	572.44
1 MONTH	35	MIN	485.17	458.79	295.96	169.99	159.05
		AVR	459.8	449.26	95.566	50.024	50.024
2 MONTH	35	MIN	547.68	503.39	133.31	47.494	38.718
		AVR	543.56	453.32	39.953	38.082	38.07

**Table 4.20** Population size 60

	NOD	AVR	100	200	500	1000	1500
1 MONTH	3	MIN	487.45	487.42	487.4	487.4	487.4
		AVR	487.41	487.4	487.4	487.4	487.4
2 MONTH	3	MIN	572.44	572.44	572.44	572.44	572.44
		AVR	572.44	572.44	572.44	572.44	572.44
1 MONTH	35	MIN	497.3	453.95	236.33	60.456	47.294
		AVR	468.59	440.12	82.891	28.813	28.813
2 MONTH	35	MIN	535.4	346.5	48.313	38.9	37.597
		AVR	530.03	168.32	38.646	37.583	37.578

**Table 4.21** Population size 80

	NOD	AVR	100	200	500	1000	1500
1 MONTH	3	MIN	487.46	487.42	487.4	487.4	487.4
		AVR	487.44	487.4	487.4	487.4	487.4
2 MONTH	3	MIN	572.44	572.44	572.44	572.44	572.44
		AVR	572.44	572.44	572.44	572.44	572.44
1 MONTH	35	MIN	483.74	434.27	253.23	102.21	101.81
		AVR	453.55	276.03	84.636	28.813	28.813
2 MONTH	35	MIN	507.33	286.23	51.043	50.954	50.954
		AVR	480	284.45	37.788	37.611	37.611

**Table 4.22** Population size 100

	NOD	AVR	100	200	500	1000	1500
1 MONTH	3	MIN	487.44	487.42	487.4	487.4	487.4
		AVR	487.41	487.4	487.4	487.4	487.4
2 MONTH	3	MIN	572.44	572.44	572.44	572.44	572.44
		AVR	572.44	572.44	572.44	572.44	572.44
1 MONTH	35	MIN	495.42	445.11	188.25	112	82.092
		AVR	481.31	405.26	84.636	28.813	28.813
2 MONTH	35	MIN	541.4	245.5	53.54	37.817	37.627
		AVR	540.58	95.345	40.266	37.62	37.611

#### 4.6. Comparison Of ABC And PSO

In Table 4.23 results of PSO, IPSO and MPSO are taken from Table 4.3, Table 4.5 and Table 3.6. Also Mse values of ABC taken from Table 4.16 to Table 4.21. In addition, both ABC and PSO based algorithms used the same fitness evaluation and the structure of the food sources and the particles are the same. ABC, PSO, IPSO and MPSO were tested with 20, 30, 40, 50, 60, 80 population sizes and each had 100, 200, 500, and 1000 maximum iterations. Table 4.23 represents averages of minimums (best found solution) for 10 runs with respect to population size and maximum iteration number.

Table 4.23 compares the results of ABC, PSO, IPSO and MPSO with respect to several iterations and population sizes. In most cases MSE of the ABC is less than PSO based

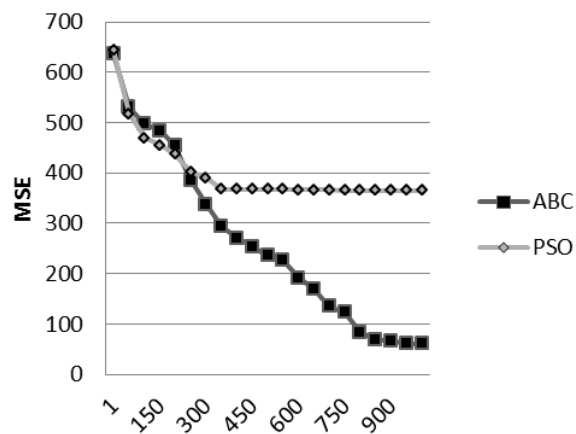
algorithms and the exceptions are shown with grey color. The difference between ABC and PSO increases with increasing iterations. In PSO based algorithms with increasing iteration size results getting better. This is because inertia weight decreases linearly through iterations. In high iteration sizes, the algorithm has better performance for global search after the local search. In 100 and 200 iterations differences are not too much, but after 500 iterations the performance of ABC is significantly better than the PSO, IPSO and MPSO. Most of the improvements are accrued until about iteration 500 in PSO. There is no substantial performance increase in higher number of iterations in this algorithm, because most of the time the PSO gets trapped in a local optimal.

**Table 4.23** Comparison of ABC with PSO based Mehtods

Iteration No	Population Size	ABC	PSO	IPSO	MPSO
1000	20	255.68	427.38	420.74	378.81
	30	221.16	402.23	385.1	395.47
	40	302.26	395.75	401.7	339.37
	50	224.02	401.23	403.82	374.54
	60	75.56	365.39	381.58	360.63
	80	99.76	261.4	258.293	371.7
500	20	345.9	402.44	402.42	421.82
	30	340.56	375.93	374.52	350.73
	40	325.85	393.68	448.37	385.5
	50	315.36	378.93	360.08	347.13
	60	270.81	337.99	340.29	350.12
	80	260.93	348.16	330.2	296.83
200	20	469.51	470.01	465.01	483.95
	30	479.36	480.55	489.98	480.65
	40	480.63	431.89	450.237	420.7
	50	450.97	401.86	388.652	375.17
	60	460.48	398.46	391.758	380.6
	80	450.27	397.09	387.96	383.85
100	20	540.93	558.81	522.87	521.3
	30	505.65	531.95	510.56	539.85
	40	502.35	505.23	513.64	481.16
	50	490.63	520.65	547.21	505.42
	60	499.54	490.48	481.41	490.49
	80	487.93	491.69	496.69	465.1

Figure 4.5 shows averages of 10 runs in each iteration for the ABC and the PSO for iteration size 1000 and population size 60. The convergence of the PSO is faster than the ABC in early iterations, but later the performance gain comes to a standstill and cannot go beyond local optimal. On the contrary, the performance of the ABC increases with

higher number of iterations. Table 4.23 represents best solution found in 10 runs for the ABC and the PSO for 1000 iteration size. ABC has better performance here, too. As well as IPSO and MPSO have the same behaviour with a little improvements. As a result, since both PSO and ABC have the same time complexity, we can conclude that ABC performs better in this problem.



**Figure 4.5** Comparison of ABC and PSO

## **Chapter 5: CONCLUSION**

In this part of study, previous parts of thesis will be discussed briefly. The aim of this study is to build a simple and effective model to forecasting next 24 hour of electricity market prices with highest possible accuracy. In feature extraction phase many features produced such as lagged prices and Mas which are highly correlated to target price. Features extracted from just price values. There are 32 features that is needed to deal with as a high dimension problem. In this case, some feature may not provide additional information for training or may become irrelevant (redundant) when using them with other features. Mentioned issue can cause rising error in training and testing of model. Therefore, feature selection is needed to provide relevant features as small number as possible. EC and NIH techniques applied in many studies in the case of feature selection and proved that they could find acceptable subset of features without excavation all search space.

Among many machine learning and statistical methods for electricity market price forecasting, ANN and SVR received more attention because of complexities of dealing with this problem and also they applied in many studies successfully. Due to advantages of SVR over ANN such as minimization of structural risk, it was selected to build a model. There are two issues must be considered in dealing with SVR to increase accuracy: first, choosing appropriate hyper parameters and second is provide suitable features specially when facing with high dimensional problems like electricity market price forecasting.

Different EC and NIH techniques applied for parameter optimization of SVR as well as feature selection which between them PSO attracted many researchers. Most of studies deal with parameter optimization and feature selection separately but recently few studies tried to combine these two in one place with GA and PSO. They claimed that simultaneous parameter optimization and feature selection could provide good solutions too rather than use different processes, also with simple model. So based on aims of this study which are build a simple and effective model, study continued with simultaneous feature selection and parameter optimization based on PSO. At first standard version of PSO implemented but the experiments showed that trapping in local optimal happen too much and performance of the model does not meet expectations. Because of mentioned reason, IPSO proposed based IBPSO which is continuous version of IBPSO, so it is

named IPSO. In IPSO, if there is no improvement in global best particle after predefined number of iterations it will reinitialize. But experiments proved that this way does not help to avoid trapping as well, besides results showed improvement about 2-3%. There another techniques is needed to overcome this problem. So MPSO method was proposed based on MBPSO which is continuous version of MBPSO. In MBPSO there two techniques to prevent stick in local optimal. One is to reinitialize global best particle as it mention before in IPSO and second is to apply small mutations to particles. Combination of these two cause 2-3% improvements in performance but outcome still has quite distance with expectations. As conclusion for PSO based methods implemented in this study, they showed a good performance in original studies but in this case they did not provide acceptable performance.

Recently ABC algorithm due to its ability to avoid trapping in local optimal received many attentions too. Also it applied in parameter optimization and feature selection problem successfully. Therefore the ABC was implemented for simultaneous feature selection and parameter optimization. Results showed that ABC naturally has ability to avoid trapping in local optimal and achieve better outcome in comparison with PSO based methods. There were different experiments with ABC such as to find better parameters for ABC and examine different datasets due to increase accuracy. Based on those experiments, this study achieved about 95% accuracy of regression model for electricity market price forecasting which is possibly one of the best results between many studies in this field.

## REFERENCES

- [1] T. W. Berrie, (1992). *Electricity Economics and Planning*, Peter Peregrinus Ltd., London.
- [2] Alamaniotis, M., Ikonomopoulos, A., Alamaniotis, A., Bargiotas, D., Tsoukalas, L.H., (2012) Day-ahead electricity price forecasting using optimized multiple-regression of relevance vector machines, 8th Mediterranean Conference on , Power Generation, Transmission, Distribution and Energy Conversion (MEDPOWER 2012), 1-5.
- [3] Shahidehpour, M., Yamin, H., Li, Z., (2002) *Market Operations in Electric Power Systems: Forecasting, Scheduling, and Risk Management*, John Wiley & Sons Inc, New York.
- [4] Contreras, J., Espinola, R., Nogales, F. J., Conejo, A. J., (2003) ARIMA models to predict next-day electricity prices, *IEEE Trans. Power Syst.*, 18(3), 1014-1020.
- [5] Conejo, A. J., Plazas, M. A., Espinola, R., Molina, A. B., (2005) Dayahead electricity price forecasting using the wavelet transform and ARIMA models, *IEEE Trans. Power Syst.*, 20(2), 1035- 1042.
- [6] Garcia R. C., Contreras J., van Akkeren M., (2005) A GARCH forecasting model to predict day-ahead electricity prices, *IEEE Trans on Power Systems*, 20(2), 867-874.
- [7] Tripathi, M. M., Upadhyay, K. G., Singh, S. N., (2008) Short-Term Load Forecasting using Generalized Regression and Probabilistic Neural Networks in the Electricity Market, *The Electricity*, 21, 24-34.
- [8] Feng, G., Xiaohong, G., Xi-Ren, C., Papalexopoulos, A., (2000) Forecasting power market clearing price and quantity using a neural network method, *IEEE Power Engineering Society Summer Meeting*, 4, 2183-2188.
- [9] Mandal, P., Urasaki, N., K. Srivastava, A., (2007) A Novel Approach to Forecast Electricity Price for PJM Using Neural Network and Similar Days Method, *IEEE Trans on Power Systems*, 22(4), 1058-1065.
- [10] Liu, D., Niu, D.X., Xing, M., NIE, Q., (2007) Day-ahead price forecast with Geneticalgorithm- optimized Support Vector Machines based on GARCH error calibration, *Automation of Electric Power Systems*, 31 (11), 31–34.
- [11] Cao, L.J., (2003) Support vector machines experts for time series forecasting, *Neurocomputing* , 51, 321–339.
- [12] Qun, Z., Wenjing, L., Liqian, D., (2006) Parameters selection for SVR based on PSO, *The Sixth World Congress on ,Intelligent Control and Automation*, 1, 2811-2814.
- [13] Haijun, C., Ahmed, M., (2010) Application of support vector regression trained by particle swarm optimization in warrant price prediction, *2nd International Conference on ,Industrial Mechatronics and Automation (ICIMA)*, 1, 358-361.
- [14] Zhang, G.P., (2000) Neural networks for classification: a survey, *IEEE Trans. Syst.*, 30 (4), 451–462.
- [15] Cheng-San, Y., Li-Yeh, C., Chao-Hsuan, K., Cheng-Hong, Y., (2008) Boolean binary particle swarm optimization for feature selection, *Evolutionary Computation*, 2093-2098.
- [16] Hongtao, Z., Hanping, M., (2009) Feature Selection for the Stored-grain Insects Based on PSO and SVM, *Knowledge Discovery and Data Mining*, 586-589.
- [17] Shih-Wei, Lin., Kuo-Ching, Y., Shih-Chieh, C., Zne-Jung, L., (2008) Particle swarm optimization for parameter determination and feature selection of support vector machines, *Expert Systems with Applications*, 35(4), 1817-1824.

- [18] Jiansheng, W., Enhong, C., (2010) A Novel Hybrid Particle Swarm Optimization for Feature Selection and Kernel Optimization in Support Vector Regression, 2010 International Conference on , Computational Intelligence and Security (CIS), 189-194.
- [19] Quan-Zhu, Y., Jie, C., Jiu-Long, Z., (2009) Simultaneous Feature Selection and LS-SVM Parameters Optimization Algorithm Based on PSO, WRI World Congress on , Computer Science and Information Engineering, 5, 723-727.
- [20] Vapnik, V., (1995) The Nature of Statistical Learning Theory, Springer, New York.
- [21] Schölkopf, B., Smola, A. J., Williamson, R., Bartlett, P. L., (2000) New support vector algorithms, Neural Computation, 12(5), 1207-1245.
- [22] Yitian, X., (2012) A rough margin-based linear support vector regression, Statistics & Probability Letters, 82(3), 528-534.
- [23] Guosheng, H., Liang, H., Hongwei, L., Kun, L., Wei, L., (2010) Grid Resources Prediction with Support Vector Regression and Particle Swarm Optimization, Third International Joint Conference on , Computational Science and Optimization (CSO), 1, 417-422.
- [24] Wang, J., Li, L., Niu, D., Tan, Z., (2012) An annual load forecasting model based on support vector regression with differential evolution algorithm, Applied Energy, 94, 65-70.
- [25] Jiang, M., Jiang, S., Zhu, L., Wang, Y., Huang, W., Zhang, H., (2013) Study on Parameter Optimization for Support Vector Regression in Solving the Inverse ECG Problem, Computational and Mathematical Methods in Medicine, 2013.
- [26] Wu, C. H., Tzeng, G. H., Lin, R. H., (2009) A Novel hybrid genetic algorithm for kernel function and parameter optimization in support vector regression, Expert Syst., 36( 3), 4725-4735.
- [27] Zhang, X., Chen, X., He, Z., (2010) An ACO-based algorithm for parameter optimization of support vector machines, Expert Syst. Appl., 37( 9), 6618-6628.
- [28] Pudil, P., Novovičová, J., (1998) Novel Methods for Feature Subset Selection with Respect to Problem Knowledge, Feature Extraction, Construction and Selection. H. Liu and H. Motoda, Springer US, 453: 101-116.
- [29] "Curse of dimensionality." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc., 14 April 2014. Web. 21 April 2014.  
<[http://en.wikipedia.org/wiki/Curse\\_of\\_dimensionality](http://en.wikipedia.org/wiki/Curse_of_dimensionality) >
- [30] Oommen, T., Misra, D., Twarakavi, N. K. C., Prakash, A., Sahoo, B., Bandopadhyay, S., (2008) An Objective Analysis of Support Vector Machine Based Classification for Remote Sensing, Mathematical Geosciences 40 (4).
- [31] Kira, K., Rendell, L.A., (1992) A practical approach to feature selection, in: Assorted Conferences and Workshops, 249-256.
- [32] Zhu, Z.X., Ong, Y.S., Dash, M., (2007) Wrapper-filter feature selection algorithm using a memetic framework, IEEE Transactions on Systems, Man, and Cybernetics, 37, 70-76.
- [33] Neshatian, K., Zhang, M., (2009) Pareto front feature selection: using genetic programming to explore feature space, in: Proceedings of the 11th Annual conference on Genetic and evolutionary computation (GECCO'09), 1027-1034.
- [34] Chen, Y., Miao, D., Wang, R., (2010) A rough set approach to feature selection based on ant colony optimization, Pattern Recognition Letters, 31, 226-233.
- [35] SyarifahAdilah, M.Y., Abdullah, R., Venkat, I., (2012) ABC algorithm as feature selection for biomarker discovery in mass spectrometry analysis, 4th Conference on , Data Mining and Optimization (DMO), 67-72.

- [36] Alba, E., Garcia-Nieto, J., Jourdan, L., Talbi, E.-G., (2007) Gene selection in cancer classification using PSO/SVM and GA/SVM hybrid algorithms, in: Proc. of the IEEE Congress on Evolutionary Computation, 284–290.
- [37] Chuang, L.Y., Chang, H.W., Tu, C.J., Yang, C.H., (2008) Improved binary PSO for featureselection using gene expression data, *Computational Biology and Chemistry*, 32 (1), 29–38.
- [38] Chuang, L.Y., Tsai, S.W., Yang, C.H., (2011) Improved binary particle swarm optimizationusing catfish effect for feature selection, *Expert Systems with Applications*, 38, 12699–12707.
- [39] Ying-Chun, G., (2009) An integrated PSO for parameter determination and feature selection of SVR and its application in STLF, *International Conference on , Machine Learning and Cybernetics*, 1, 359-364.
- [41] Kennedy, J., Eberhart, R.C., Shi, Y., (2001) *Swarm Intelligence*, Morgan Kaufmann Publishers Inc., San Francisco, CA.
- [42] M. Vieira, S., F. Mendonça,L., J. Farinha, G., M.C. Sousa, G., (2013) Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients, *Applied Soft Computing*, 13(8), 3494-3504.
- [43] Shi, Y., Eberhart, R.C., (1998) Parameter selection in particle swarm optimization. In *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, pages 591-600.
- [44] Shi, Y., Eberhart, R.C., (1999) Empirical study of particle swarm optimization, In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, 1945-1950.
- [45] Clerc, M., Kennedy, J., (2002) The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evolutionary Computation*, 6(1), 58-73.
- [46] van den Bergh, F., (2002) An analysis of particle swarm optimizers. PhD thesis, University of Pretoria, South Africa.
- [47] Trelea, I. C., (2003) The particle swarm optimization algorithm: convergence analysis and parameter selection. *Inf. Process. Lett.*, 85(6), 317-325.
- [48] Kennedy, J., Eberhart, R.C., (1997) A discrete binary version of the particle swarm algorithm. In *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, 4104-4109.
- [49] Parsopoulos, K. E., Vrahatis, M. N., Recent approaches to global optimization problems through particle swarm optimization. *Natural Computing: an international journal*, 1(2-3),n235-306.
- [50] Angeline, P.J., (1998) Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. In *Proceedings of the VII Conference on Evolutionary Programming*, 601-610.
- [51] Eberhart R. C., Shi, Y., (2000) Comparing inertia weights and constriction factors in particle swarm optimization. In *Proceedings of the Congress on Evolutionary Computing*, 84-89.
- [52] LI, Ai., (2004) Cooperative and optimal algorithm of multiparticles”,*Journal of Fudan University (Nature Science)* 43(5).
- [53] Shi, Y., Eberhart, R., (1998) A modified particle swarm optimizer, *IEEE World Congress on Computational Intelligence., Evolutionary Computation Proceedings*, 69-73.

- [54] Kennedy, J., Eberhart, R., (1995) A new optimizer using particle swarm theory, Proceedings of the Sixth International Symposium on , Micro Machine Human Science, 39-43.
- [55] Sheng-Fa, Y., Fu-Lei, C., (2007) Fault diagnostics based on particle swarm optimization and support vector machines, Mechanical Systems and Signal Processing, 21(4), 1787-1798.
- [56] Hostload data set. Available at <[http://cs.uchicago.edu/lyang/ Loadl](http://cs.uchicago.edu/lyang/Loadl)>, Jul 19, 2003.
- [57] Shian, Z., Lingzhi, W., (2010) Support Vector Regression Based on Particle Swarm Optimization for Rainfall Forecasting, Third International Joint Conference on , Computational Science and Optimization (CSO), 2, 484-487.
- [58] Vapnik, V., Golowich, S., Smola, A., (1997) Support vector method for function approximation, regression estimation and signal processing, Advance in neural information processing system, 9, 281–287.
- [59] Ruijin, L., Hanbo, Z., Grzybowski, S., Lijun, Y., (2011) Particle swarm optimization-least squares support vector regression based forecasting model on dissolved gases in oil-filled power transformers, Electric Power Systems Research, 81(12), 2074-2080.
- [60] Rehman, A.U., Khanum, A., Shaukat, A., (2013) Hybrid Feature Selection and Tumor Identification in Brain MRI Using Swarm Intelligence, 11th International Conference on , Frontiers of Information Technology (FIT), 49-54.
- [61] Haralick R.M., Shanmugam, K., Dinstein, I., (1973) Textural Features for Image Classification, IEEE Trans. on Systems, Man and Cybernetics, 3(6), 610 - 621.
- [62] Linyi, L., (2011) Feature Selection for Residential Area Recognition in High Resolution Images Based on Particle Swarm Optimization, Remote Sensing, Environment and Transportation Engineering (RSETE), 357-360.
- [63] Chmulik, M., Jarina, R., Kuba, M., (2012) On objective feature selection for affective sounds discrimination, ELMAR, 199-202.
- [64] Bäck, T., Fogel, D. B., Michalewicz, Z., (1997) The handbook of evolutionary computation, Oxford University Press.
- [65] Vieira, S.M., Sousa, J.M.C., Runkler, T.A., Two cooperative ant colonies for feature selection using fuzzy models, Expert Systems with Applications, 37, 2714–2723.
- [66] Cismondi, F., Horn, A.L., Fialho, A.S., Vieira, S.M., Reti, S.R., Sousa, J.M.C., Finkelstein, S., (2012) Multi-stage modeling using fuzzy multi-criteria feature selection to improve survival prediction of ICU septic shock patients, Expert Systems with Applications, 39 (16), 12332–12339.
- [67] Zhan, Z.-H., Zhang, J., Li, Y., Chung, H.-H., (2009) Adaptive particle swarm optimization, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 39 (6), 1362–1381.
- [68] Yuan, L., Zhao, Z.-D., (2007) A modified binary particle swarm optimization algorithm for permutation flow shop problem, in: Proc. of the International Conference on Machine Learning and Cybernetics, 2, 902–907.
- [69] Farinha, G.J., Vieira, S.M., Mendonc, L.F., Sousa, J.M., (2011) Optimization of fuzzy models using a novel PSO algorithm: application to medical databases, Proc. Of the 18th World Congress of the International Federation of Automatic control, 9023-9028.

- [70] Vieira, S.M., Mendonc, L.F., Farinha, G.J., Sousa, J.M.,(2012) Metaheuristics for feature selection: application to sepsis outcome prediction, the World Congress on Computational Intelligence, 2395–2402.
- [71] Asuncion, A., Newman, D., (2007) UCI machine learning repository, URL [www.archive.ics.uci.edu/ml](http://www.archive.ics.uci.edu/ml)
- [72] Huang, C.-L., Wang, C.-J., (2006) A GA-based feature selection and parameters optimization for support vector machines, *Expert Systems with Applications*, 31 (2), 231–240.
- [73] Smola, A. J, (1998) *Learning with kernels*, Technical University of Berlin, Berlin.
- [74] Lee, S., Soak, S., Oh, S., Pedrycz, W., Jeon, M., (2008) Modified binary particle swarm optimization, *Progress in Natural Science*, 18 (9), 1161–1166.
- [75] Karaboga D., (2005) An idea based on honey bee swarm for numerical optimization. Technical report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.
- [76] Karaboga, D., Basturk, B., (2008) On the performance of artificial bee colony (ABC) algorithm. *Appl Soft Comput*, 8, 687-697.
- [77] Karaboga, D., Basturk, B., (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *Journal of Global Optimization*, 39, 459–471.
- [78] Suguna, N., Thanushkodi, K.G., (2011) An Independent Rough Set Approach Hybrid with Artificial Bee Colony Algorithm for Dimensionality Reduction, *American Journal of Applied Sciences*, 8 (3), 261-266.
- [79] Suguna, N., Thanushkodi, K.G., (2010) A novel Rough Set Reduct Algorithm for Medical Domain based on Bee Colony Optimization, *Journal of Computing*, 2(6),49-54.
- [80] Hsieh, T.J., Yeh, W.C., (2011) Knowledge discovery employing grid scheme least squares support vector machines based on orthogonal design bee colony algorithm. *IEEE Trans. Syst. Man Cybern.*, 41(5), 1–15.
- [81] Yusof, Y., Mustaffa, Z., (2011) Optimizing LSSVM using ABC for non-volatile financial prediction, *Journal of Applied Sciences Research*, 7(11), 549.
- [82] Bonabeau, E., Dorigo, M., Theraulaz, G., (1999) *Swarm intelligence: from natural to artificial systems*. Oxford University Press, Inc, New York.
- [83] Millonas, M. M., (1994) *Swarms, phase transitions, and collective intelligence*, *Artificial life III*, 417–445.
- [84] Dorigo, M., Maniezzo, V., Colorni, A., (1991) Positive feedback as a search strategy. Technical Report 91-016, Politecnico di Milano, Italy
- [85] Kennedy, J., Eberhart, R. (1995) Particle swarm optimization, *IEEE international conference on neural networks*, 1942–1948.
- [86] Karaboga, D., Akay, B., (2009) A comparative study of Artificial Bee Colony algorithm, *Applied Mathematics and Computation*, 214(1), 108-132.
- [87] Tereshko, V., Loengarov, A., (2005) Collective decision-making in honeybee foraging dynamics, *Computing and Information Systems Journal*, 9 (3), 1-7.
- [88] Schiezero, M., Pedrini, H., (2013) Data feature selection based on Artificial Bee Colony algorithm, *EURASIP Journal on Image and Video Processing*, (1), 1-8.
- [89] Mustaffa, Z., Yusof, Y., (2012) A hybridization of enhanced artificial bee colony-least squares support vector machines for price forecasting, *J. Comput. Sci.*, 8, 1680-1690.
- [90]. Chih-Ming, Hsu., (2013) Optimizing the Design of a TIR Lens Using SVR, VIKOR, and the Artificial Bee Colony Algorithm, *IAENG International Conference on*

Industrial Engineering, 824-830.

[91] Opricovic, S., (1998) Multicriteria Optimization in Civil Engineering. Belgrade: Faculty of Civil Engineering.

[92] Sulaiman, M. H., Mustafa, M. W., Shareef, H., Abd. Khalid, S. N., (2012) An application of artificial bee colony algorithm with least squares support vector machine for real and reactive power tracing in deregulated power system, International Journal of Electrical Power & Energy Systems, 37(1), 67-77.

[93] Pelkmans, K., Suykens, J., Gestel, T., Brabanter, J., Lukas, L., Hamers, B., LSSVMLab: A Matlab/C toolbox for least squares support vector machines. ESATSISTA, K.U. Leuven, Leuven, Belgium; 2002.

[94] Zimmerman, R., Murillo-Sanchez, C., Thomas, R., MATPOWER: steady-state operations, planning, and analysis tools for power systems research and education. IEEE Trans Power Syst, 26(1), 12-9.

[95] Chang, C.-C., Lin, C.-J., (2002) Training  $\nu$ -support vector regression: Theory and algorithms, Neural Computation, 14(8):1959-1977.

[96] Vahidinasab, V., Jadid, S., Kazemi, A., (2008) Day-ahead price forecasting in restructured power systems using artificial neural networks, Electric Power Systems Research, 78(8), 1332-1342.

[97] Amjady, N., Daraeepour, A., (2009) Design of input vector for day-ahead price forecasting of electricity markets, Expert Systems with Applications, 36(10), 12281-12294.

[98] Xiaohui, H., (2010) Particle Swarm Optimization,  
<http://www.swarmintelligence.org/>

[99] "Swarm intelligence." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc., 7 March 2014. Web. 21 April 2014.

[http://en.wikipedia.org/wiki/Swarm\\_intelligence](http://en.wikipedia.org/wiki/Swarm_intelligence)

[100] "Moving average." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc., 16 April 2014. Web. 21 April 2014. [http://en.wikipedia.org/wiki/Moving\\_average](http://en.wikipedia.org/wiki/Moving_average)

[101] Chang, C.C., Lin, C.J., LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>

## **RESUME**

### ALIASGHAR KAVIAN

CELL : 05319214770

MAIL : [aliasghar.kavian@gmail.com](mailto:aliasghar.kavian@gmail.com)

Date of Birth : 21/05/1983

Place of Birth : IRAN

Marital St. : Single

### Education

2010-continues: Marmara University, Computer Engineering Department Master of Science

2001-2005: Islamic Azad University, Computer Engineering department.