REFERENCES

[1] Fels, M., Special issue devoted to measuring energy savings, the Princeton scorekeeping method (PRISM), Energy and Buildings, 1986, 9, 1-2.

[2] Kissock, J.K., A methodology to measure retrofit energy savings in commercial buildings, Doctoral Dissertation, Department of Mechanical Engineering, Texas A&M University, 1993.

[3] Krarti, M. J., Kreider, D., Cohen, P., Curtiss, Prediction of energy saving for building retrofits using neural networks, ASME Journal of Solar Energy Engineering, 1988, 120, 31-35.

[4] Dhar, A., Reddy, T.A., Claridge, A. D. E., Series model to predict hourly heating and cooling energy use in commercial buildings with outdoor temperature as the only weather variable, Journal of Solar Energy Engineering, 1999, 121, 47-53.

[5] Dong B., Lee S.E., Sapar M.H.,A holistic utility bill analysis method for baselining whole commercial building energy consumption in Singapore, Energy and Buildings, 2005, 37 (2), 167-172.

[6] Dong B., Chen C., Lee S.E., Applying support vector machines to predict building energy consumption in the tropics, Energy and Buildings, 2005, 37(5), 545-553.

[7] Lokmanhekim, M., Henninger, R. H., Computerized energy requirements analysis and heating/cooling load calculations of buildings. ASHRAE J, 1972, 14(4), 25-33.

[8] Adrian, S., Brhan, T., Akram, R., Cooling and heating loads in residential buildings in Jordan, Energy Build, 1997, 26 (2),137-43.

[9] Lao Stephen, T. H., Deng S. M., An evaluation of the rules of thumb for estimating cooling load for office buildings, Trans Hong Kong Inst Eng, 2001, 8(3), 58-9.

[10] Bojic M., Yik F., Wan K., Burnett, J., Investigations of cooling loads in high-rise residential buildings in Hong Kong, Strojniski Vestnik.J Mech. Eng., 2001, 47(8), 491.

[11] Probst Oliver., Cooling load of buildings, Appl. Energy, 2004, 77(2), 171-186.

[12] BojiMilorad., Yik F., Cooling energy evaluation for high-rise residential buildings in Hong Kong, Energy Build, 2005, 37(4), 345-51.

[13] Ansari F. A., Mokhtar A. S., Abbas K. A., Adam N. M., A simple approach for building cooling load estimation. Am J Environ Sci., 2005, 1(3), 209-20.

[14] Chou S. K., Chang W. L., Large building cooling load and energy use estimation. Int J Energy Res., 1997, 21(2), 169-83.

[15] Greg J. S., Robert A. K., A correlation method for predicting monthly and annual cooling loads in direct gain passive solar heated buildings. Energy Convers Manage., 1989, 29(3), 175-87.

[16] Bauer M., Scartezzini J. L., A simplified correlation method accounting for heating and cooling loads in energy-efficient buildings, Energy Build, 1998, 27(2), 147-54.

[17] Sodha M. S., Kaur B., Kumar A., Bansal N. K., A comparison of the admittance and fourier methods for predicting heating/cooling loads, Solar Energy, 1986, 36(2), 125-7.

[18] Ben-NakhiAbdullatif E., Mahmoud M. A., Cooling load prediction for buildings using general regression neural networks, Energy Converse Manage, 2004, 45(13), 2127-41.

[19] Mui K. W., Wong L. T., Cooling load calculations in sub tropical climate, Build Environ, 2007, 42(7), 2498-504.

[20] Lam J. C., Wan K. K. W., Tsang C. L., Yang L., Building energy efficiency in different climates, Energy Convers Manage, 2008, 49, 2354-66.

[21] Thevenard D., Haddad K., Ground reflectivity in the context of building energy simulation, Energy Build, 2006, 38, 972-80.

[22] Eskin N., Türkmen H., Analysis of annual heating and cooling energy requirements for office buildings in different climates in Turkey, Energy Build, 2008, 40, 763-73.

[23] González Pedro A., Zamarreño Jesús M., Prediction of hourly energy consumption in buildings based on a feedback artificial neural network, Energy Build, 2005, 37(6), 595-601.

[24] Yuan Cheng., Minh-Tu Cao., Accurately predicting building energy performance using evolutionary multivariate adaptive regression, 2014, 22,178-188.

[25] Rishee K., Kevin M., Patricia J. Culligan., John E., Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy, 2014, 123, 168-179.

[26] Chunlei Zeng., Changchun Wu., LiliZuo., Bin Zhang, Xingqiao Hu, Predicting energy consumption of multiproduct pipeline using artificial neural networks, 2004, 66, 791-798.

[27] Dong, B., Cao, C., Lee S. E., Applying Support Vector Machines to Predict Building Energy Consumption in Tropical Region, Energy and Buildings, 2005, 37, 122-26.

[28] Vapnik V., The nature of statistical learning, New York: Springer, 1995, 2, 20-32.

[29] LibSVM "README" tutorial

[30] Hsu., Chang, C-W., Lin, C. C., A practical Guide to Support Vector Classification, Department of Computer Science, National Taiwan University.

[31] Qiong Li., Applying support vector machine to predict hourly cooling load in the building, 2009, 86, 2249-2256.

[32] Dew Point, http://en.wikipedia.org/wiki/Dewpoint.

[33] Kreider, J. F., and Wang, X. A., (1992). Improved Artificial Neural Networks for Commercial Building Energy Use Prediction. Proceedings of the 1992 ASME-JSES-KSES International Solar Energy Conference, Maui, HI, Apr. 1992, 5-9, 361-366.

[34] Hour, Z., Lian, Z., An Application of support vector machines in cooling load prediction, School of Mechanical and Electrical Engineering, Shenzhen Polytechnic and Institute of Refrigeration and Cryogenics Shanhai Jiao Tong University, 2009.

[35] Knebel, D.E., Simplified Energy Analysis Using the Modified Bin Method, American Society of Heating and Air-Conditioning Engineers, Atlanta, GA, 1983.

[36] Meng, Q., Cai, J., Yoshino, H., Mochida, A., Predicting Hourly Cooling Load in the Building: A Comparison of Support Vector Ma- chine and Different Artificial Neural Networks, Energy Conversion and Management, 2009, 5, 90-96.

[37] Cao, L. J., Francis, E. H., Support vector machine with adaptive parameters in financial time series forecasting, IEEE Transactions on Neural Networks, 2003, 14(6), 1506-1518.

[38] ASHRAE. Handbook of Fundamentals, American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc., Atlanta, GA, 2001.

[39] ASHRAE. ASHRAE GUIDELINE 14: Measurement of Energy and Demand Savings. Atlanta: American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc, 2002.

[40] Akbari, H., Heinemeier, K., LeConiac, P., Flora, D. An Algorithm to Disaffregate Commercial Whole-Building Electric Hourly Load into End Uses, Summer Study on Energy Efficiency in Buildings, 1988,10,13-26.

[41] Fels, M. F., and Keating, K. M., Measurement of Energy Savings from demand-side management programs in U.S. electric utilities, Annual Review Energy Environ, 1993, 18, 57-88.

[42] Reddy. T. A., Saman, N. F., Claridge, D. E., Haberl, J. S., Turner, W. D., Chalifoux, A., Base lining Methodology for Facility-Level Monthly Energy Use – Part 1: Theoretical Aspects, ASHRAE Transactions, 1997, 103, 2.

[43] Katipamula S., T. A. Reddy, Claridge D. E., Multivariate Regression Modeling, ASME Journal of Solar Energy Engineering, 1998, 115, 177-184.

[44] MATLAB 6.5 user's manual. The Math Works, Inc

[45] Chang, C.-C., and Lin, C.-J. (2001). LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[46] Chen, Z.Q., Nelson, R.M. and Ashlock, D.A. (2003). Comparison of methods for predicting monthly psot-retrofit energy use in buildings, ASHRAE Transactions, 2003,109, 449-459.

[47] Kreider, J. F., Claridge, D. E., Curtiss, P., Dodier, R., Haberl, J. S., Krarti, M., Building Energy Use Prediction and System Identification Using Recurrent Neural Networks, ASME Journal of Solar Energy Engineering, 1995, 117,161-166.

[48] IPMVP. (2001). International Performance Measurement and Verification Protocol, DOI/GO-102001-1187, Washington D.C.: Department of Energy.

[49] Eto, J., on using degree-days to account for the effects of weather on annual energy use in office buildings. Energy and Building, 1998, 9, 5-18.

[50] Vapnik, V. N., Golowich, S. E., and Smola, A. J., Support vector method for function approximation, regression estimation and signal processing, Adv. Neural Information Processing Syst. 1996, 9, 281-287.

[51] Bowerman, B. L., and O'Connell, R. T., Linear Statistical Models: An Applied Approach, 1990, Second Edition, Duxbury Press, Belmont, California.

[52] Draper, N., and H. Smith., Applied Regression Analysis, 1981, 2, 506.

[53] Day, A. R., Karayiannis, T. G., A New Degree-day Model for Estimating Energy Demand in Buildings, Building Services Engineering research and Technology, 1999 20(4), 171-178.

[54] Arens, E. A., Carroll, W. L., 1978. Geographical variation in the heating and cooling requirements of a typical single family house and correlation of these requirements to degree day. NBS-BBS 116. Gaithersburg, MD: National Bureau of Standards.

[55] Kissock, J. K., and Fels, M., An assessment of PRISM's reliability for commercial buildings. Proceedings of National Energy Program Evaluation Conference, Chicago, IL, August, 1995.

[56] Day, A. R., and Karayiannis, T. G., (1999), Identification of the Uncertainties in Degree-day Based Energy Estimates, Building Services Engineering research and Technology, 1999, 20(4), 165-172.

[57] Katipamula S., Reddy, T. A., Claridge D. E., Multivariate Regression Modeling,ASME, Journal of Solar Energy Engineering, 1998, 115, 177-184.

[58] Sondereggger, R. A., Baseline model for utility bill analysis using both weatherand non-weather-related variables, ASHRAE transactions, 1998, 104, 859-870.

[59] Bronson, D., Hinchey, S., Haberl, J. S., O'Neal, D. L., Claridge, D. E., A Procedure for calibrating the DOE-2 simulation program to non-weather dependent measured loads, ASHRAE Transactions, 1992, 98(1), 636-652.

[60] Katipamula, S., Claridge, D.E., Use of Simplified System Models to MeasureRetrofit energy savings, Journal of Solar Energy Engineering, 1993, 115, 57-68.

[61] Kyung-Jin Jang, Eric B. Bartlett, Ron M. N., Measuring retrofit energy savings using auto associative neural networks. ASHRAE Transactions, 1996,102(2), 412-418.

[62] Knebel, D. E., Simplified energy analysis using the modified bin method, American Society of Heating, Refrigerating and Air-Conditioning Engineers, Atlanta, GA, 1983.

[63] Cherkassky, V., Ma, Y., Practical selection of SVM parameters and noise estimation for SVM regression, Neural Networks, 2004, 17, 113-126.

[64] Mukherjee, S., Osuma, E., Girosi, F., Nonlinear prediction of chaotic timeseries using support vector machine, in Proceedings of IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing VII (NNSP’97), 1997, 21, 511-520.

[65] Katipamula, S., Claridge, D.E., Use of simplified system models to measure retrofit energy savings, Journal of Solar Energy Engineering, 1993,115, 57-68.

APPENDIX A: AN OVERVIEW OF STATISTICAL LEARNING THEORY

APPENDIX A: AN OVERVIEW OF STATISTICAL LEARNING THEORY

Vapnik (1995) presents all the machine learning problems as the following:
Given a set of data points $((x_1,y_1),(x_2,y_2),\ldots,(x_1,\ y_1)$ ($x_i \in X \subseteq R^2, y_i \in Y \subseteq R, l$ is the number of data points, for regression estimation and density estimation and $y_i \in Y \subseteq N$ for pattern recognition) randomly and independently generated from an unknown probability distribution *p*(x,y), find a function *f*(x,a) that has the minimal risk function (A.1).

$$R(f) = \int_{x,y} L9y, f(x,a))p(x,y)dxdy \qquad \text{(A.1)}$$

Where a is the parameter of f(x,a). R(*f*) is called the generalization error or the expected test error. It is a measure of the generalization performance of *f* (x,a) . L(y, *f*(x,a)) is called the loss function. It is a measure of the deviations between the actual values and the estimated values on the data points generated from *p*(x,y)
As *p*(x,y) is unknown, traditional methods attempt to estimate *f* (x,a) by minimizing the empirical risk function:

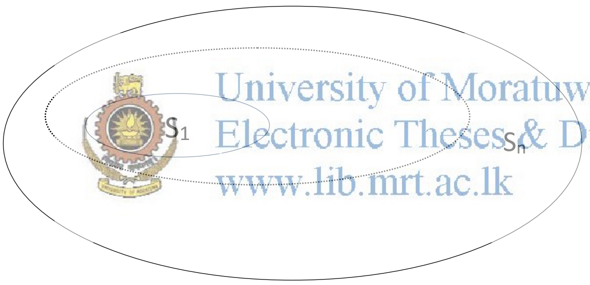
$$R_{emp}(f) = \frac{1}{l}\sum_{i=1}^{l} L\,(x,a)) \qquad \text{(A.2)}$$

$R_{emp}(f)$ is called the empirical error. That is, is estimated by training samples. Empirical Risk Minimization (ERM) principle is to minimize the generalization error by minimizing the empirical error $R_{emp}(f)$. Traditional neural networks utilize this principle.

However, because of the limited number of *l*, sometimes $R_{emp}(f)$ cannot estimate R(*f*) well. As described in the statistical learning theory, and have the following relationship:

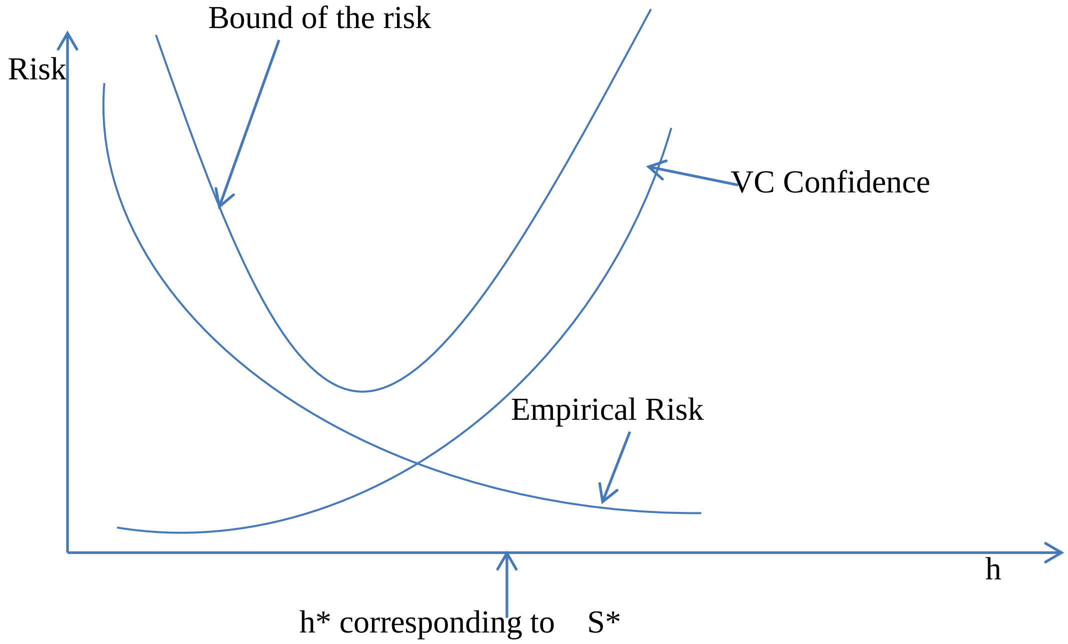
$$R(f) \leq R_{emp}(f) + \Omega(\frac{l}{h}) \qquad \text{(A.3)}$$

Where h is called the Vapnik-Chervonenkis (VC) dimension. It is a measure of the capacity of *f(*x,a), which means that the ability of *f*(x,a) to learn any training data point without error. $\Omega\left(\frac{l}{h}\right)$ is called the confidence interval, a decreasing function of $\frac{l}{h}$, which is the ration of the number of training samples into the VC dimension of the estimator. Equation (A.3) shows that the value of R(*f*) depends both on $R_{emp}(f)$ and $\Omega(\frac{l}{h})$. Hence, $R_{emp}(f)$ can accurately estimate $R(f)$ only when $\Omega(\frac{l}{h})$ is small enough. Because of this, Vapnik developed the Structural Risk Minimization (SRM) principle. The SRM principle is: one defines a nested structure, $S_1 \subset S_2 \subset \ldots \subset S_m \ldots$, as shown in Appendix B.1, on the set of functions $S = \{f(,a), a \in \Lambda\}$ with their VC-dimensions satisfying $h_1 \subset h_2 \subset \ldots \subset h_m \ldots$, and then chooses the structure element S_k with the minimal upper bound of the generalization error R(*f*).



Appendix A.1: A structure on the set of functions is determined by the nested subsets of functions.

The objective of SRM principle is to estimate *f*(x,a) by minimizing both the empirical error $R_{emp}(f)$ and the confidence interval $\Omega(\frac{l}{h})$, as shown in Appendix A.2. It defines a trade-off between the quality of the approximation of the given data and the complexity of the approximating function.



Appendix A.2: The bound on the risk is the sum of the empirical risk and of the confidence interval. The smallest bound of the risk is achieved on some appropriate element of the structure (Source: Vapnik, 1995)

APPENDIX B: AN INTRODUCTION TO LIBSVM 2.6 PROGRAM

APPENDIX B: AN INTRODUCTION TO LIBSVM 2.6 PROGRAM

Libsvm is a simple, easy-to-use, and efficient software for SVM classification and regression. Libsvm 2.6 was developed by Chih-Chung Chang and Chih-Jen Lin in 2001. It can solve C-SVM classification, nu-SVM classification, one-class-SVM, epsilon-SVM regression, and nu-SVM regression. It also provides an automatic model selection tool for C-SVM classification.

Libsvm 2.6 is available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

The format of training and testing data file is:
<label><index1>:<value1><index2>:<value2> ...

<label> is the target value of the training data. For classification, it should be an integer which identifies a class (multi-class classification= is supported). For regression, it's any real number. For one-class SVM, it's not used so can be any number. <index> is an integer starting from 1, <value> is a real number. The labels in the testing data file are only used to calculate accuracy or error. If they are unknown, just fill this column with a number.

'svm-train' Usage

Usage: svm-train [options] training_set_file [model_file]
options:
-s svm_type : set type of SVM (default 0)
0 -- C-SVC
1 -- nu-SVC
2 -- one-class SVM
3 -- epsilon-SVR
4 -- nu-SVR
-t kernel_type : set type of kernel function (default 2)

0 -- linear: u'*v

1 -- polynomial: (gamma*u'*v + coef0)^degree

2 -- radial basis function: exp(-gamma*|u-v|^2)

3 -- sigmoid: tanh(gamma*u'*v + coef0)

-d degree : set degree in kernel function (default 3)

-g gamma : set gamma in kernel function (default 1/k)

-r coef0 : set coef0 in kernel function (default 0)

-c cost : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)

-n nu : set the parameter nu of nu-SVC, one-class SVM, and nu-SVR (default 0.5)

-p epsilon : set the epsilon in loss function of epsilon-SVR (default 0.1)

-m cachesize : set cache memory size in MB (default 40)

-e epsilon : set tolerance of termination criterion (default 0.001)

-h shrinking: whether to use the shrinking heuristics, 0 or 1 (default 1)

-b probability_estimates: whether to train an SVC or SVR model for probability estimates, 0 or 1 (default 0)

-wi weight: set the parameter C of class i to weight*C in C-SVC (default 1)

-v n: n-fold cross validation mode

The k in the -g option means the number of attributes in the input data.

option -v randomly splits the data into n parts and calculates cross validation accuracy/mean squared error on them.

‘svm-predict’ Usage

Usage: svm-predict [options] test_filemodel_fileoutput_file

-b probability estimates: whether to predict probability estimates, 0 or 1 (default 0); one-class SVM not supported yet

model_file is the model file generated by svm-train.

test_file is the test data you want to predict.

svm-predict will produce output in the output_file.

(Source: Chang and Lin, 2001)

APPENDIX C: MAT LAB CODES FOR PROGRAM

APPENDIX C: MAT LAB CODES FOR PROGRAM

```
devidePoint=600;
shift=10;
% output data vection creation for svmtrain
% normalize the output
output = (ElectricalConsumption(:,3)
min(ElectricalConsumption(:,3)))/(max(ElectricalConsumption(:,3))
min(ElectricalConsumption(:,3)));

TrainOutput = output(shift+1:devidePoint+1,:);  TestOutput =
output(devidePoint+2:end,:);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% this section belongs to input data matrix creation
 ‘%normalize inputs
Humidity_norm = (Humidity(:,4)-min(Humidity(:,4)))/(max(Humidity(:,4)-
min(Humidity(:,4)));
MaxTemperature_norm = (MaxTemperature(:,4)-
min(MaxTemperature(:,4)))/(max(MaxTemperature(:,4))-
min(MaxTemperature(:,4)));
SolarRadiation_norm = (SolarRadiation(:,4)-
min(SolarRadiation(:,4)))/(max(SolarRadiation(:,4))-min(SolarRadiation(:,4)));
%moving average input & Outputs
%outputMa1 = transpose(tsmovavg(transpose(output), 'e', shift));
 %outputMa2 = transpose(tsmovavg(transpose(MaxTemperature_norm ), 'e',
shift));
%outputMa3 = transpose(tsmovavg(transpose( Humidity_norm), 'e', shift));
 %outputMa4 = transpose(tsmovavg(transpose(SolarRadiation_norm ), 'e', shift));
%create time delay input
      timeDelayInput1   = [output(1:end)];
```

```
    timeDelayInput11  = [output(1);output(1:end-1)];
    timeDelayInput111 = [output(1:2);output(1:end-2)];

   %timeDelayInput2   = [output(1:end)];
   %timeDelayInput22  = [output(1);output(1:end-1)];
   %timeDelayInput222 = [output(1:2);output(1:end-2)];

   %timeDelayInput3   = [outputMa3(1:end)];
   %timeDelayInput33  = [outputMa3(1);outputMa3(1:end-1)];
   %timeDelayInput333 = [outputMa3(1:2);outputMa3(1:end-2)];

   %timeDelayInput4   = [outputMa4(1:end)];
   %timeDelayInput44  = [outputMa4(1);outputMa4(1:end-1)];
   %timeDelayInput444 = [outputMa4(1:2);outputMa4(1:end-2)];
 input =
[timeDelayInput1,timeDelayInput11,timeDelayInput111,MaxTemperature_norm,Hu
midity_norm,SolarRadiation_norm];%,outputMa2,outputMa3,outputMa4];
   %devide input vector for training and testing
    TrainInput = input(shift:devidePoint,:);
    TestInput = input(devidePoint+1:end-1,:);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% train svm model and set parameters
% Usage: model = svmtrain(training_label_vector, training_instance_matrix,
'libsvm_options');

% libsvm_options:
% -s svm_type : set type of SVM (default 0)
%       0 -- C-SVC
%       1 -- nu-SVC
%       2 -- one-class SVM
%       3 -- epsilon-SVR
```

```
%        4 -- nu-SVR
% -t kernel_type : set type of kernel function (default 2)
%        0 -- linear: u'*v
%        1 -- polynomial: (gamma*u'*v + coef0)^degree
%        2 -- radial basis function: exp(-gamma*|u-v|^2)
%        3 -- sigmoid: tanh(gamma*u'*v + coef0)
%        4 -- precomputed kernel (kernel values in training_instance_matrix)

% -d degree : set degree in kernel function (default 3)
% -g gamma : set gamma in kernel function (default 1/num_features)
% -r coef0 : set coef0 in kernel function (default 0)
% -c cost : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)
% -n nu : set the parameter nu of nu-SVC, one-class SVM, and nu-SVR (default 0.5)
% -p epsilon : set the epsilon in loss function of epsilon-SVR (default 0.1)
% -m cachesize : set cache memory size in MB (default 100)
% -e epsilon : set tolerance of termination criterion (default 0.001)
% -h shrinking : whether to use the shrinking heuristics, 0 or 1 (default 1)
% -b probability_estimates : whether to train a SVC or SVR model for probability
estimates, 0 or 1 (default 0)
% -wi weight : set the parameter C of class i to weight*C, for C-SVC (default 1)
% -v n : n-fold cross validation mode
% -q : quiet mode (no outputs)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   %crossvalidation = svmtrain(TrainOutput,TrainInput,'-m 800 -v 5')
   %K1 = [(1:630), 'K']; % include sample serial number as first column
   %model = svmtrain(label_vector, K1, '-t 4');
   %matlab> [predict_label, accuracy, dec_values] = svmpredict(label_vector, K1,
model); % test the training data
```

```
ForecastingModel = svmtrain(TrainOutput,TrainInput,'-s 3  -t 2  -g 0.25 -c 1 -p 0.125');

 %svm-scale -l -1 -u 1 -s range train > train.scale
 %svm-scale -r range test > test.scale
%Scale each feature of the training data to be in [-1,1]. Scaling
%factors are stored in the file range and then used for scaling the
%test data.

% svm-train -s 0 -c 5 -t 2 -g 0.5 -e 0.1 data_file
%Train a classifier with RBF kernel exp(-0.5|u-v|^2), C=10, and
%stopping tolerance 0.1.
%svm-train -s 3 -p 0.1 -t 0 data_file
%Solve SVM regression with linear kernel u'v and epsilon=0.1 in the loss function.
%svm-train -c 10 -w1 1 -w2 5 -w4 2 data_file
%Train a classifier with penalty 10 = 1 * 10 for class 1, penalty 50 = 5 * 10 for class 2, and penalty 20 = 2 * 10 for class 4.
%svm-train -s 0 -c 100 -g 0.1 -v 5 data_file
%Do five-fold cross validation for the classifier using the parameters C = 100 and gamma = 0.1
%svm-train -s 0 -b 1 data_file
%svm-predict -b 1 test_file data_file.model output_file
%Obtain a model with probability information and predict test data with probability estimates

%svm_parameter(svm_type=0,kernel_type=2,gamma=1,cache_size=40,eps=0.001,C=1,nr_weight=0,shrinking=1)
    %crossvalidation = svmtrain(TrainOutput,TrainInput,'-m 800 -v 5')
    %/svm-train -c 2 -g 2 svmguide1.scale
    %./svm-predict svmguide1.t.scale svmguide1.scale.model svmguide1.t.predict
    %./svm-train -s 4 -t 2 -g .1 -c 120 TrainFile.txt ModelFile.txt
    %. /svm-predict TestFile.txt ModelFile.txtOutputFile.txt.
```

```
    %./svm-train svmguide1
    %./svm-predict svmguide1.t svmguide1.model svmguide1.t.predict
% test svm model
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[predictionP, accuracyP, decvalueP] = svmpredict(TestOutput,TestInput,
ForecastingModel,'-b 0');
    predictionP;
    accuracyP;
    decvalueP;
    TestOutput;
p1=plot(predictionP);
set(p1,'Color','red')
hold on
p2=plot(TestOutput);
set(p2,'Color','black')
%%%%%%%%%%%%%%%%%%%Whether data%%%%%%%%%%%%
%p2=plot(Humidity_norm)
%set(p2,'Color','red')
%hold on

%p3=plot(MaxTemperature_norm)
%set(p3,'Color','black')
%hold on

%p5=plot(SolarRadiation_norm)
%set(p5,'Color','blue')
%hold on

%%%%%%%%%%%%%%MSE & R^2, Nsv data%%%%%%%%%%%%%%%
%For constant gamma = 0.1 and varing C values
%p2=plot(X,Y)
```

```
%set(p2,'Color','red')
%p3=plot(X,Z)
%set(p3,'Color','black')

%p5=plot(X,xxa)
%set(p5,'Color','blue')
%hold on
```