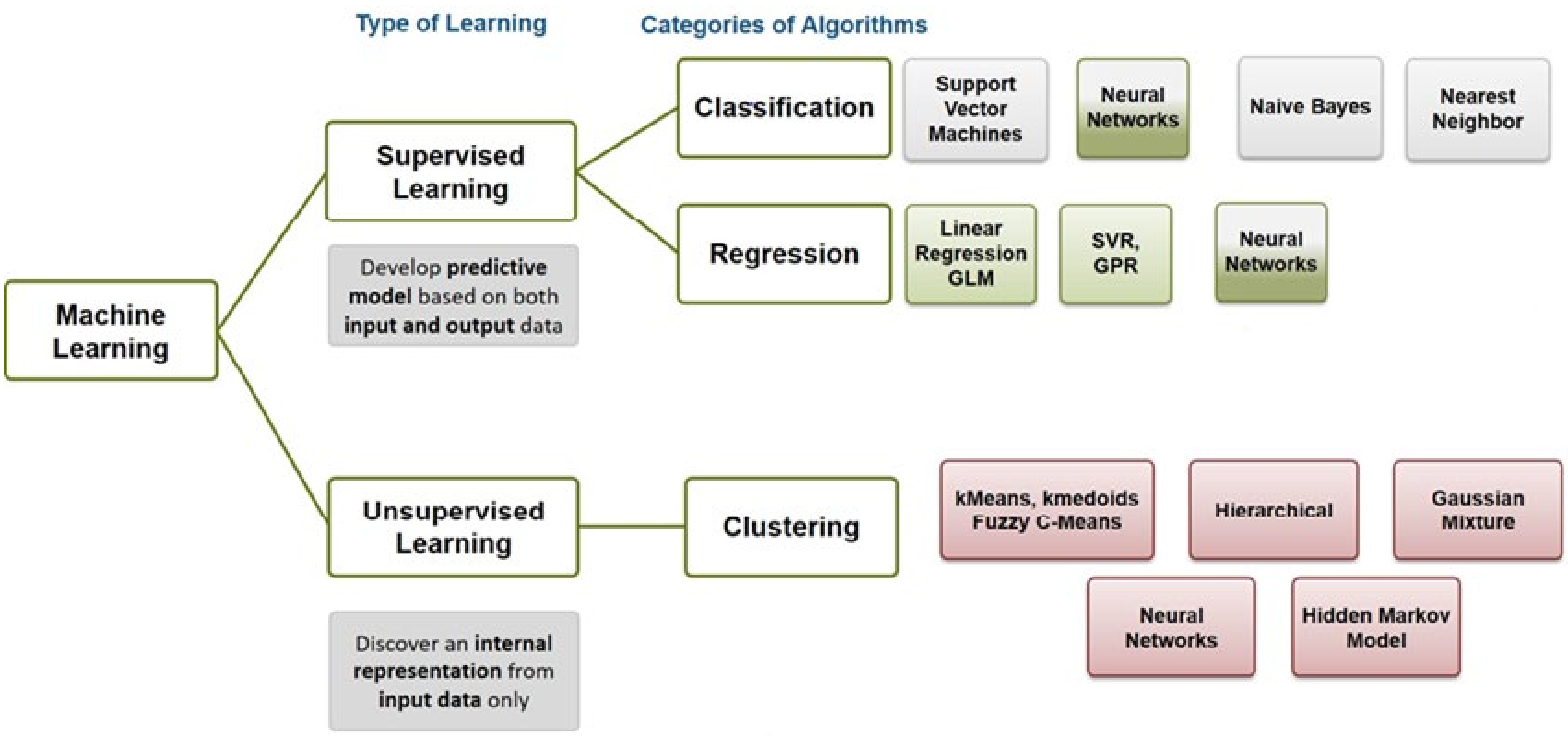
Machine Learning
Practical -1
Support Vector Machine (SVM)

Lecturer:

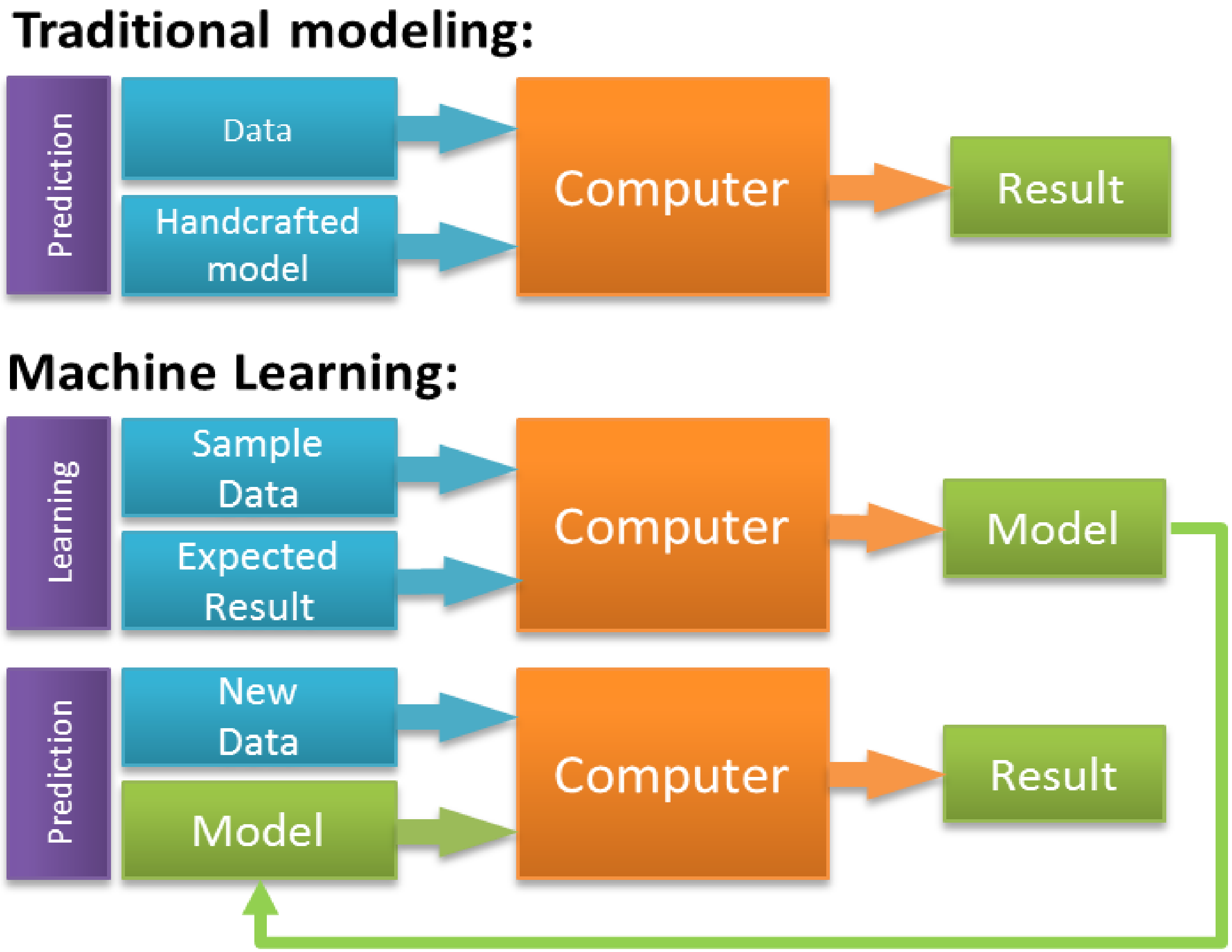
Dr Hamid Khayyam (Australia)

Email: hamid.khayyam@rmit.edu.au

Recap

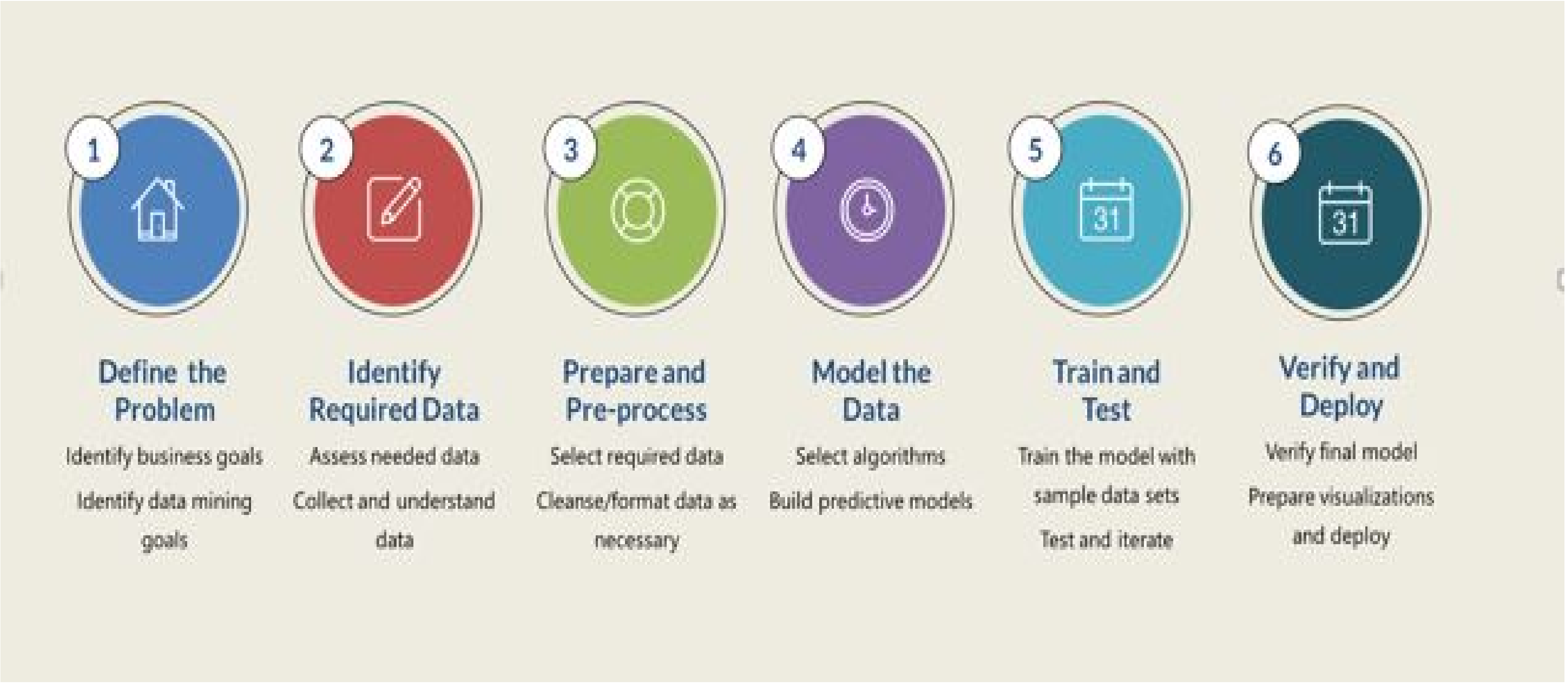

Type of Learning
Categories of Algorithms
Machine Learning
Supervised Learning
Develop predictive model based on both input and output data
Unsupervised Learning
Discover an internal representation from input data only
Classification
Support Vector Machines
Neural Networks
Naive Bayes
Nearest Neighbor
Regression
Linear Regression GLM
SVR, GPR
Neural Networks
Clustering
kMeans, kmedoids Fuzzy C-Means
Hierarchical
Gaussian Mixture
Neural Networks
Hidden Markov Model

Recap



Comparison of analytical and machine learning models

General steps in machine learning



Recap

- ## Access and load the data

- **load filename.mat or load('filename.mat')**

- **Read Microsoft Excel spreadsheet file**

- **Read comma-separated value (CSV) file or text files**

- ## Data pre-processing

- **Find Missing data (NaN,missing)**
  - Replace or and Ignore missing Data

Recap

- **Find and replacing outliers**
  - Replace or and Ignore outliers
- **Data standardization**

- **Data normalization**

- **Feature extraction and reduction**
- **Removing redundant or irrelevant features(inputs)**

- **Combining features**

- **Creating new features**

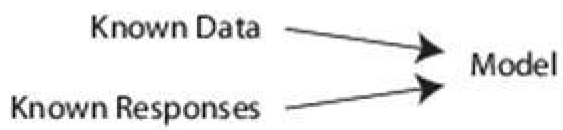
Steps of An Application of SVM

1. Data pre-processing (check for missing data ,standardization)

2. Model development and training

3. Mdl = fitcsvm(X, Y) (Classification) % Mdl is the developed model

   Mdl = fitrsvm(X, Y) (Regression)



3. Simulation (prediction)

   label= predict(Mdl,X) (classification) % label : predicted labels

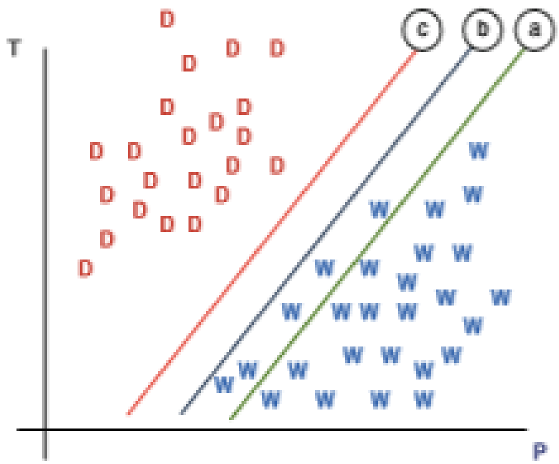
   Y_predicted= predict(Mdl,X) (Regression) %Y_predicted is the predicted responses

4. Post-processing

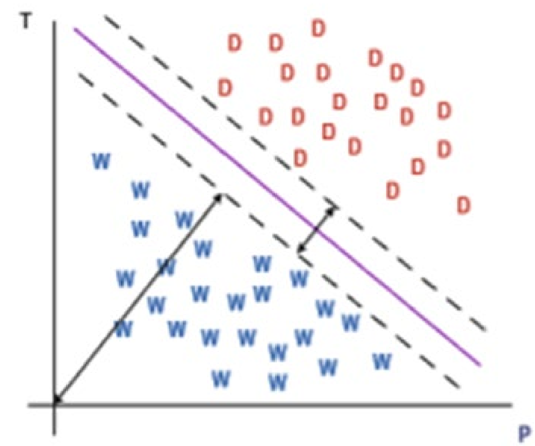
   - MSE,RMSE,R (Regression)

Support vector Machine (SVM)

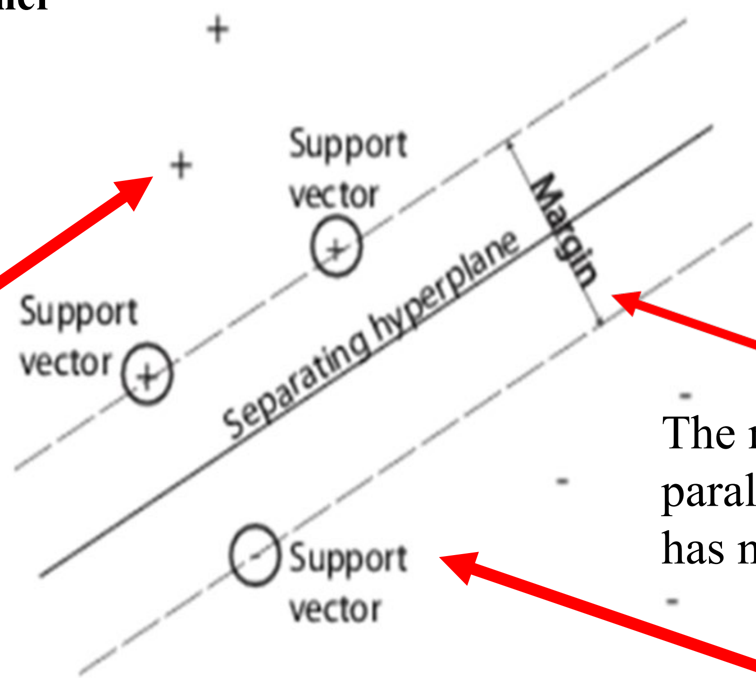
- Support vector machine (SVM) analysis is a popular machine learning tool for classification and regression.
- SVM is useful specially when the dataset is small.
- SVM builds a hyperplane in between datasets to indicate which class it belongs to. The best hyperplane for an SVM means the one with the largest margin between the two classes.
- SVM is less prone to overfitting when compared to ANN.
- ANNs can suffer from multiple local minima, the solution to an SVM is global and unique.
- The version of SVM for regression is support vector regression (SVR).



(a) wrong classifier, (b) an unstable classifier, and (c) a stable classifier



Classification in SVM



The maximum width of the slab parallel to the hyperplane that has no interior data points.

+ : data points of type 1
– : data points of type –1

The data points nearest to the hyperplane

Support Vector Machine (SVM) (cont.)

- **Separable data (Hard margin)**

The equation of a hyperplane :

$$W^{\mathrm{T}}X + b = 0$$

X: variables in the decision space.

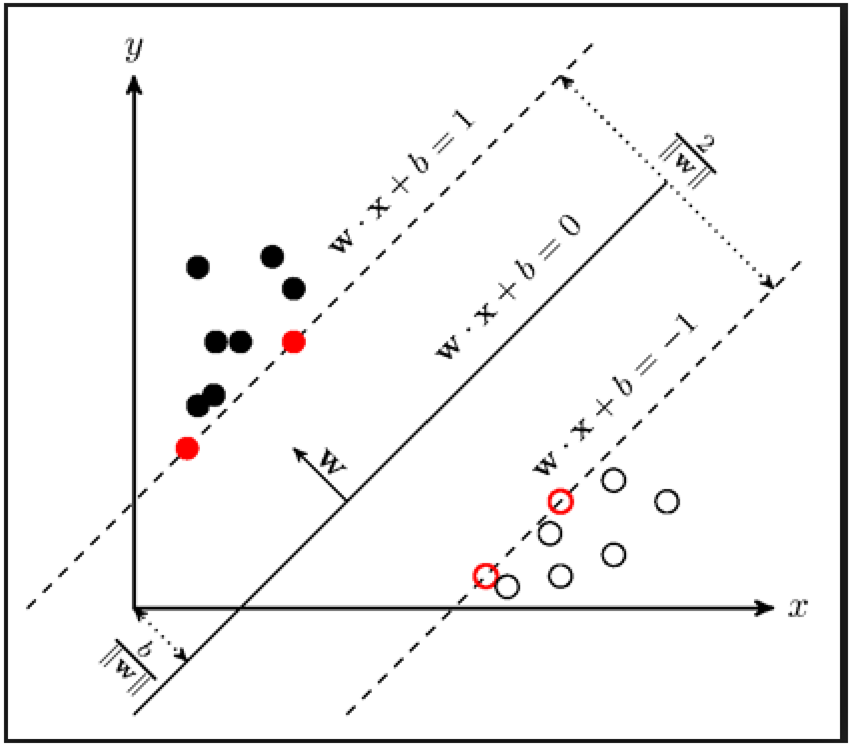
W and b are the parameters of the classifier.

The Equation of marginal lines :

$$W^{\mathrm{T}}X + b = 1$$

$$W^{\mathrm{T}}X + b = -1$$

Support vector Machine (SVM) (cont.)



$$D = 2d = \frac{2}{\|W\|}$$

Training (solving) hard-margin problem in Matlab:

$$\text{Minimize} \quad L = \frac{1}{2} W^{\mathrm{T}} W$$

Subject to

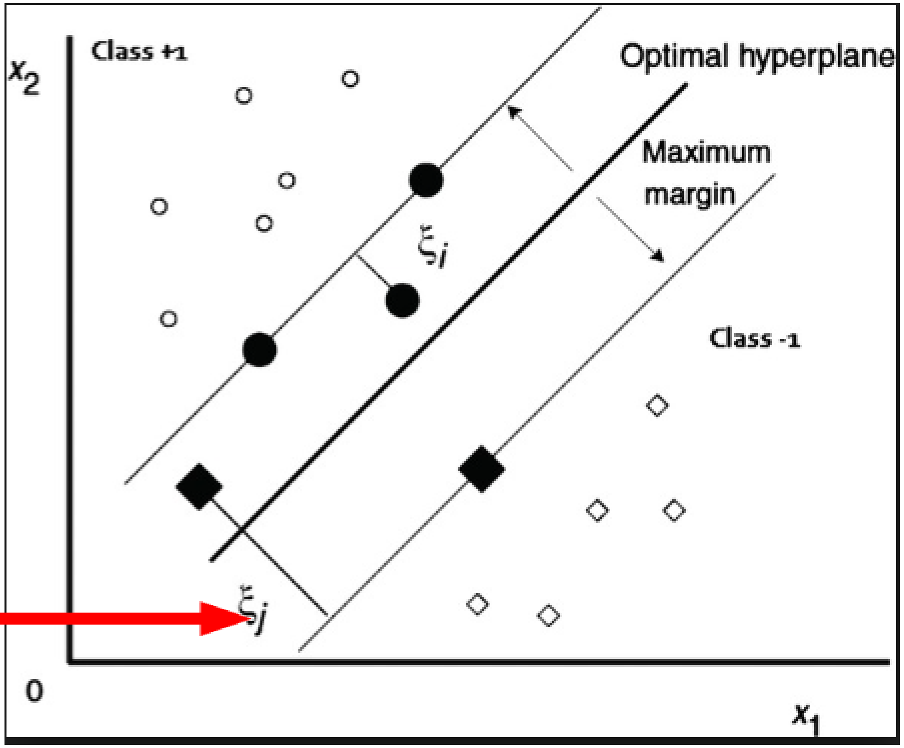
$$y\left(W^{\mathrm{T}} X + b\right) - 1 \geq 0$$

$$\text{if} \quad y = 1 \quad \text{then} \quad W^{\mathrm{T}} X + b \geq 1$$
$$\text{if} \quad y = -1 \quad \text{then} \quad W^{\mathrm{T}} X + b \leq -1$$

Support vector Machine (SVM) (cont.)

- **Non separable data (soft-margin)**

.



Error

- **Training(solving) Soft-margin problem in Matlab :**

$$\text{Minimize} \quad \frac{1}{2}W^{T}W + C\sum_{i=1}^{n}\xi_i \quad i = 1, \ldots, n$$

$$\begin{aligned} \text{Subject to :} \quad & y_i\left(W^{T}X + b\right) \geq 1 - \xi_i, && \forall i \in \{1, \ldots, n\} \\ & \xi_i \geq 0, && \forall i \in \{1, \ldots, n\} \end{aligned}$$

Support Vector Machine (SVM) (cont.)

Solution (Lagrangian multiplier) :

$$\text{Minimize} \quad L = \frac{1}{2}W^{\mathrm{T}}W - \sum_i \alpha_i \left[y\left(W^{\mathrm{T}}X + b\right) - 1\right] \quad i = 1, \ldots, n$$

α : the multiplier of the constraint.

Primal problem of SVM method

$$\begin{cases} \dfrac{dL}{dW} = 0 \Rightarrow W - \sum_i \alpha_i y_i x_i \Rightarrow W = \sum \alpha_i y_i x_i \\ \dfrac{dL}{db} = 0 \Rightarrow \sum_i \alpha_i y_i = 0 \end{cases}$$

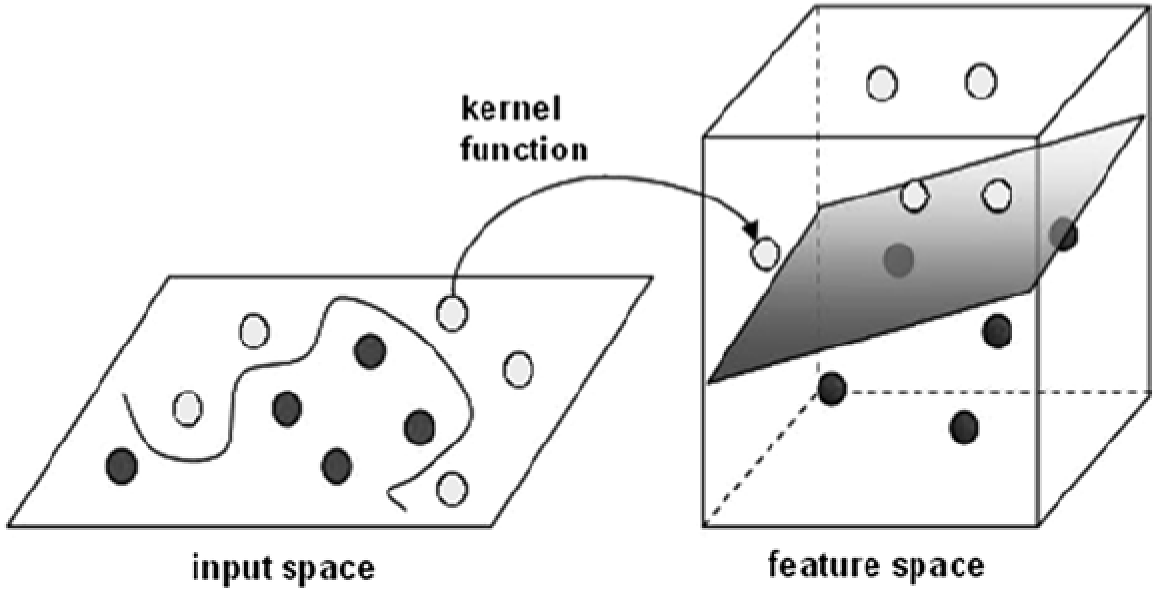
Dual problem of SVM method

$$\text{Maximize} \quad L_D = -\frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^{\mathrm{T}} x_j + \sum_i \alpha_i \quad i = 1, \ldots, n$$

$$\text{Subject to} \quad \sum_i \alpha_i y_i = 0 \quad \alpha_i \geq 0$$

Support Vector Machine (SVM) (cont.)

- **Kernel Trick Nonlinear SVM)**



Training (solving) problem with Kernel function in Matlab:

$$\text{Maximize} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j . k(x_i, x_j)$$

$$\text{Subject to :} \quad \alpha_i \geq 0, \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

$$k(x_i, x_j) = (\phi(x_i).\phi(x_j))$$

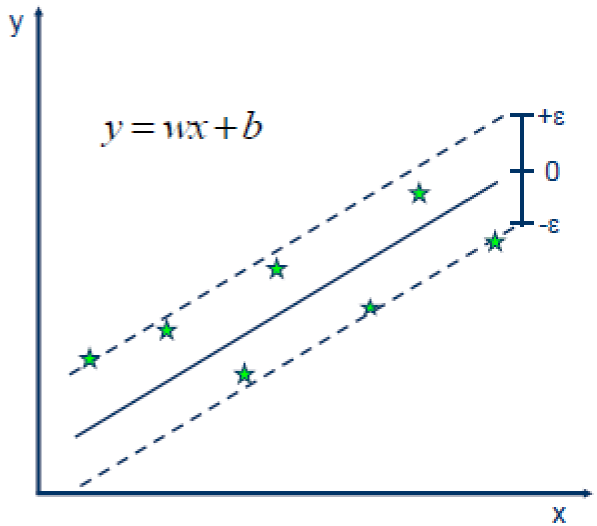
Support Vector Machine (SVM) (cont.)

- **Kernel functions**
- Linear $$k(x_i x_j) = x_i^{\mathrm{T}} x_j$$
- Polynomial $$k(x_i x_j) = (\gamma \; x_i^{\mathrm{T}} x_j + r)^d, \quad \gamma > 0$$
- RBF(Radial basis function) $$k(x_i x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \quad \gamma > 0$$

where, γ, r, and d are *kernel* parameters.

Support Vector Machine (SVM) (cont.)

- **Support Vector Machine - Regression (SVR)**



- Solution:

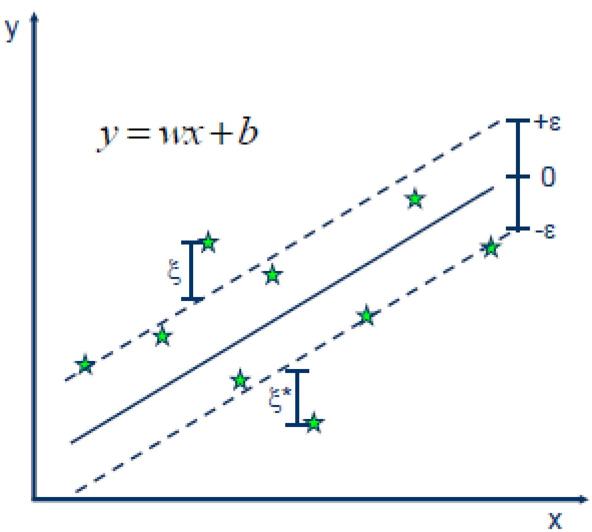
$$\min \frac{1}{2}\|w\|^2$$

Hard-margin solution

- Constraints:

$$y_i - wx_i - b \le \varepsilon$$
$$wx_i + b - y_i \le \varepsilon$$

ε: **Margin of tolerance**



- Minimize:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right)$$

- Constraints:

$$y_i - wx_i - b \le \varepsilon + \xi_i$$
$$wx_i + b - y_i \le \varepsilon + \xi_i^*$$
$$\xi_i, \xi_i^* \ge 0$$

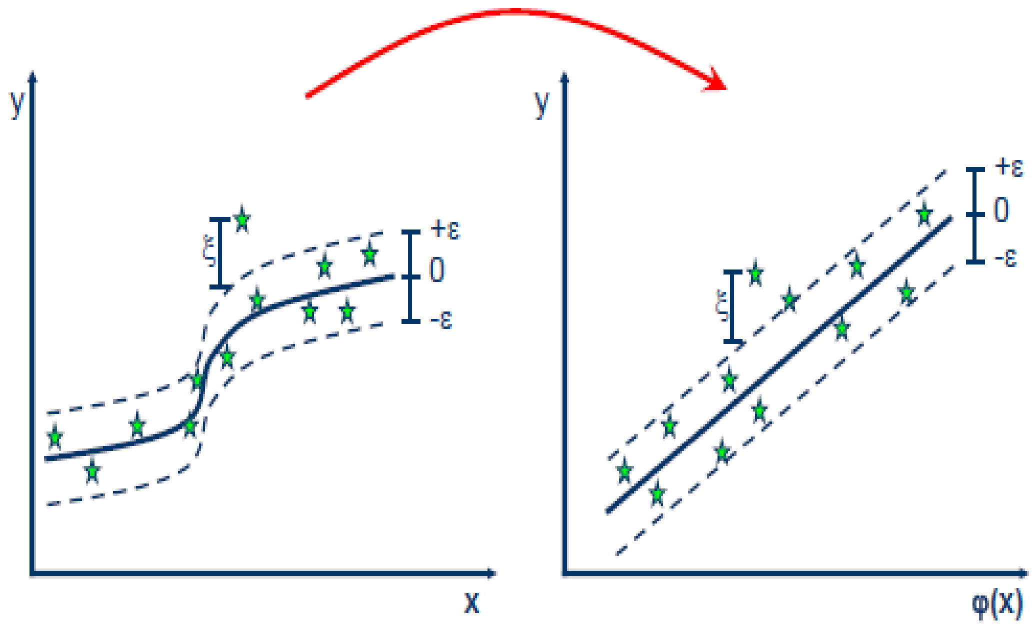
Soft-margin solution

Linear SVR: $y = \sum_{i=1}^{N}\left(\alpha_i - \alpha_i^*\right)\cdot\left\langle x_i, x\right\rangle + b$

Support Vector Machine (SVM) (cont.)

- **Nonlinear SVR**

$$y = \sum_{i=1}^{N} \left(\alpha_i - \alpha_i^*\right) \cdot K\left(x_i, x\right) + b$$



Support Vector Machine model: fitcsvm

fitcsvm

Trains a two-class (binary) classification on a low- through moderate-dimensional predictor data set.

- Use Sequential minimal optimization(SMO), Iterative Single Data Algorithms(ISDA), or L1 minimization algorithm for training of the model (Quadratic programming to minimize objective function)

Syntax

```
Mdl = fitcsvm(x, t)
```

Description

x :matrix of predicators(Input)

t :vector of class labels (Output)

Example :

```
Mdl = fitcsvm(x,t);
```

Support Vector Machine model: fitcsvm (cont.)

fitcsvm additional options:

- 'Standardize': false | true (Default: false) % Standardize data
- 'Solver': 'ISDA' | 'L1QP' | 'SMO' (Default: SMO) % Solver for objective functions

- KernelFunction

  'gaussian' or 'rbf': Gaussian or Radial Basis Function (RBF) kernel

  'linear': Linear kernel (default)

  'polynomial': Polynomial kernel % Use 'PolynomialOrder', q, to specify a polynomial kernel of order q.

- 'PolynomialOrder': positive integer (Default:3)
- 'KernelScale': 1 (default) | 'auto' | positive scalar % gamma in RBF kernel
- 'BoxConstraint': positive scalar (Default:1) % C ,the cost of misclassification
- 'OptimizeHyperparameters': 'none' (default) | 'auto' | 'all'

Support Vector Machine model: fitcsvm (cont.)

'HyperparameterOptimizationOptions' % To optimize the parameters (e.g. box constraint ,epsilon)

- 'optimizer' % The optimizer used to optimize the parameters
  - 'bayesopt' :Use Bayesian optimization.
  - 'gridsearch' :use grid search
  - 'randomsearch' — Search at random
- ShowPlots: true or false (Default :true)

Example :

```
mdl=fitcsvm(x,t,'Standardize',true,'solver','ISDA',...
'OptimizeHyperparameters','all','HyperparameterOptimizati
onOptions',struct('optimizer','gridsearch','ShowPlots',fa
lse)); % x is input ,t is output
```

Support vector machine model: predict

- **Predict**

Predict labels using trained model (Mdl)

Syntax:

ylabel= predict(Mdl,x) (classification)   % ylabel is predicted labels

Description:

x:Input Matrix

Mdl= developed and trained SVM model

ylabel : a vector of predicted class labels for matrix x, based on the trained SVM model Mdl.

Example:

```
y_predicted=predict(mdl,X);
```

Example of classication-1: fitcsvm

This dataset is consist of :
Input (x) - a 351x34 matrix
Output (t) - categorical response " b" , "g "

```
clear; clc;
load ionosphere.mat % An example of a built-in data set in Matlab
rng(1);  % random number generation to reproduce results
mdl = fitcsvm(X,Y); %Train the model;X is the input;Y is the output
y_expected=predict(mdl,X); %y_expected is the predicted output
table( Y( 10: 20),y_expected( 10:20), 'VariableNames',...
    {' TrueLabel',' PredictedLabel'})
```

11×2 table

TrueLabel	PredictedLabel
'b'	'b'
'g'	'g'
'b'	'b'
'g'	'g'
'b'	'g'
'g'	'g'
'b'	'b'
'g'	'g'
'b'	'b'
'g'	'g'
'b'	'b'

Support vector machine model: fitcecoc

- **fitcecoc**

Fit multiclass models for support vector machines

Syntax :

$\boxed{\text{Mdl = fitcecoc(x,t)}}$

Description :

x : input

t : class labels

Example:

```
clear;clc;

load fisheriris.mat

x = meas;

t = species;

Mdl = fitcecoc(x,t) ;
```

Support Vector Regression (SVR): fitrsvm

- **fitrsvm**

trains an SVM regression model for low- through moderate-dimensional predictor data sets.

  - Use Sequential minimal optimization(SMO), Iterative Single Data Algorithms(ISDA), or L1 minimization algorithm for training of the model (Quadratic programming to minimize objective function)

Syntax :

```
Mdl = fitrsvm(x,t);
```

Description:

x: Input data

t : output data

Mdl: Developed and trained SVR Model

Support Vector Regression (SVR): fitrsvm (cont.)

fitrsvm additional options:

- 'Standardize': false | true (Default:false)
- 'Solver': 'ISDA' | 'L1QP' | 'SMO' (Default: SMO)
- KernelFunction:
  - 'gaussian' or 'rbf': Gaussian or Radial Basis Function (RBF) kernel
  - 'linear': Linear kernel (default)
  - 'polynomial': Polynomial kernel. % Use 'PolynomialOrder', q, to specify a polynomial kernel of order q.
- 'BoxConstraint': positive scalar (Default:1) % C the cost of wrong prediction
- 'KernelScale': 1 (default) | 'auto' | positive scalar % gamma in RBF kernel
- 'OptimizeHyperparameters': 'none' (default) | 'auto' | 'all'

Support Vector Regression (SVR): fitrsvm (cont.)

'Epsilon' : Half the width of epsilon-insensitive band
'HyperparameterOptimizationOptions' `% To optimize the parameters (e.g. box constraint ,epsilon )`

- 'Optimizer' `% The optimizer used to optimize the parameters`
  - 'bayesopt' :Use Bayesian optimization.
  - 'gridsearch' :use grid search
  - 'randomsearch' — Search at random
- ShowPlots: true or false (Default :true)

Example :

```
mdl=fitrsvm(x,t,'Standardize',true,'solver','ISDA',...
'OptimizeHyperparameters','all','HyperparameterOptimizat
ionOptions',struct('optimizer','gridsearch','ShowPlots',
false)); % x is input ,t is output
```

Support Vector Regression: predict

- **Predict**

Predict output using trained model (Mdl)

Syntax:

y= predict(Mdl,x) (Regression)

Description:

X:Input Matrix

Mdl= trained SVM model

y : predicted output based on the trained SVM model Mdl and matrix x .

Example :

```
y=predict(mdl,x);
```

Example of Support Vector Regression-1: fitrsvm

```
clear;clc;
rng(1); % random number generation to reproduce results
Filename='SVR1.xlsx';
Sheetread='x';
Input1='A1:M252';
Sheetread1='t';
output1='A1:A252';
Input=xlsread(Filename,Sheetread,Input1); %Read Microsoft Excel
Target=xlsread(Filename,Sheetread1,output1 );
x=Input;
t=Target;
mdl=fitrsvm(x,t,'Standardize',true); %standardize the data
yfit=predict(mdl,x); % prediction based on the developed SVR model and x as the input
```

Example of Support Vector Regression-1: fitrsvm (cont.)

```
table(t(40:50,:),yfit(40:50,:),'VariableNames',{'ObservedV
alue',' PredictedValue'}) % show 40th to 50th data in
output and predicted output

RMSE_training=sqrt(sum((yfit-t).^2)/numel(t)); % Calculate
RMSE for data
```

```
ans =

  11×2 table

    ObservedValue    PredictedValue
    _____________    ______________

    32.6             31.481
    34.5             37.671
    32.9             34.106
    31.6             33.802
      32             25.864
     7.7             10.269
    13.9             10.729
    10.8             7.9351
     5.6             9.0699
    13.6             16.759
       4             5.9059
```

RMSE training :4.2602

Example of Support Vector Regression-2: fitrsvm

```
clear;clc;

rng(1); % random number generation to reproduce results

Filename='SVR2.xlsx';

Sheetread='x';

Input1='A1:A94';

Sheetread1='t';

output1='A1:A94';

Input=xlsread(Filename,Sheetread,Input1); %Read Microsoft Excel

Target=xlsread(Filename,Sheetread1,output1 );

x=Input;

t=Target;
```

```
mdl=fitrsvm(x,t,'Standardize',true); %standardize the data and use linear kernel to develope and model the data

yfit=predict(mdl,x); % prediction based on the developed SVR model and x as the input

table(t(20:30,:),yfit(20:30,:),'VariableNames',{'ObservedValue',' PredictedValue'}) % show 20th to 30th data in output and predicted output

RMSE_training=sqrt(sum((yfit-t).^2)/numel(t)); % Calculate RMSE for data
```

```
ans =

  11×2 table
```

ObservedValue	PredictedValue
9.8589	8.7503
9.6876	8.6457
9.4722	8.5412
9.2283	8.4367
8.9701	8.3322
8.7099	8.2277
8.4579	8.1231
8.2217	8.0186
8.0065	7.9141
7.8153	7.8096
7.6494	7.7051

RMSE training :1.9942

Example of Support Vector Regression-3: fitrsvm

```
clear;clc;

rng(1);

• Filename='SVR3.xlsx'; % SVR3.xlsx has been used .

Sheetread='Sheet1';

Input1='A1:B89';

output1='C1:C89';

Input=xlsread(Filename,Sheetread,Input1); %Read
Microsoft Excel

Target=xlsread(Filename,Sheetread,output1 );

Sheetread1='Sheet2';

Input2='A1:B11';
```

Example of Support Vector Regression-3: fitrsvm (cont.)

```
Target2 ='C1:C11';

Inputnew=xlsread(Filename,Sheetread1,Input2);

Targetnew=xlsread(Filename,Sheetread1,Target2 );

x=Input;

t=Target;

xnew=Inputnew;

tnew=Targetnew;

x=fillmissing(x,'spline'); %fill in the missing input
data

t= fillmissing(t,'spline'); %fill in the missing
output data
```

Example of Support Vector Regression-3: fitrsvm (cont.)

```
mdl=fitrsvm(x,t,'Standardize',true); %standardize the data and use linear kernel to develope and model the data

yfit=predict(mdl,x); % prediction based on the developed SVR model and x as the input

RMSE=sqrt(sum((yfit-t).^2)/numel(t)); % Calculate RMSE for training data

table(t(60:70,:),yfit(60:70,:),'VariableNames',{'ObservedValue',' PredictedValue'}) % show 60th to 70th data in output and predicted output

ynew=predict(mdl,xnew); % prediction based on new data

RMSE=sqrt(sum((ynew-tnew).^2)/numel(ynew)); % Calculate RMSE for new data
```

Example of Support Vector Regression-3: fitrsvm (cont.)

11×2 table

ObservedValue	PredictedValue
26.5	26.211
20	21.763
13	15.464
19	21.642
19	22.758
16.5	17.312
16.5	11.827
13	15.017
13	16.64
13	16.498
28	25.359

RMSE training :3.5404

RMSE Testing :7.7213

Example of Support Vector Regression-4(Without kernel): fitrsvm

```
clear;clc;

rng(1);

Filename='SVR4.xlsx';

Sheetread='Sheet1';

Input1='A1:H72';

output1='I1:I72';

Input=xlsread(Filename,Sheetread,Input1); %Read Microsoft
Excel

Target=xlsread(Filename,Sheetread,output1 );

x=Input;

t=Target;

Sheetread1='Sheet2';

Input2='A1:H3';

Target2 ='I1:I3';
```

Example of Support Vector Regression-4(without kernel): fitrsvm (cont.)

```
Inputnew=xlsread(Filename,Sheetread1,Input2);

Targetnew=xlsread(Filename,Sheetread1,Target2 );

xnew=Inputnew;

tnew=Targetnew;

mdl=fitrsvm(x,t,'Standardize',true); %standardize the data
and use linear kernel to develop and model the data

yfit=predict(mdl,x); % prediction based on the developed
SVR model and x as the input

RMSE_training=sqrt(sum((yfit-t).^2)/numel(t)); % Calculate
RMSE for data

ynew=predict(mdl,xnew); % prediction based on new data


table(tnew(:),ynew(:),'VariableNames',{'ObservedValue_Newd
ata',' PredictedValue_newdata'}) % show data in output and
predicted output
```

Example of regression-4(without kernel): fitrsvm (cont.)

```
RMSE_testing=sqrt(sum((ynew-tnew).^2)/numel(tnew)); % Calculate RMSE for new data

Errorpercentage=((ynew-tnew)./tnew)*100; % Calculate error percentage for tnew and ynew
```

11×2 table

ObservedValue	PredictedValue
514	513.96
518	517
517	516.09
517	517.04
515	514.48
511	511.61
511	512.02
516	512.24
515	512.21
514	512.38
515	515.57

Example of regression-4(without kernel): fitrsvm (cont.)

3×2 table

ObservedValue_Newdata	PredictedValue_newdata
495	495.46
498	497.53
498	497.03

RMSE_training =1.9569

RMSE_testing =0.6765

Errorpercentage

3x1 double

	1	2	3	4
1	0.0934			
2	-0.0951			
3	-0.1942			
4				
5				
6				

SVM and SVR for large data-sets

- **fitclinear (classification)**

trains a linear SVM model for binary classification on a high-dimensional data set .

Syntax:

```
Mdl = fitclinear(x,t);
```

Description:

x:input

t:class labels

- **fitrlinear (regression)**

Fit a linear SVR model to high-dimensional data

Syntax:

```
Mdl = fitrlinear(x,t);
```

Description:

x:input

t:output