The machine learning framework

$$y = f(\mathbf{x})$$

output

prediction function

Image feature

Training: given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function f by minimizing the prediction error on the training set

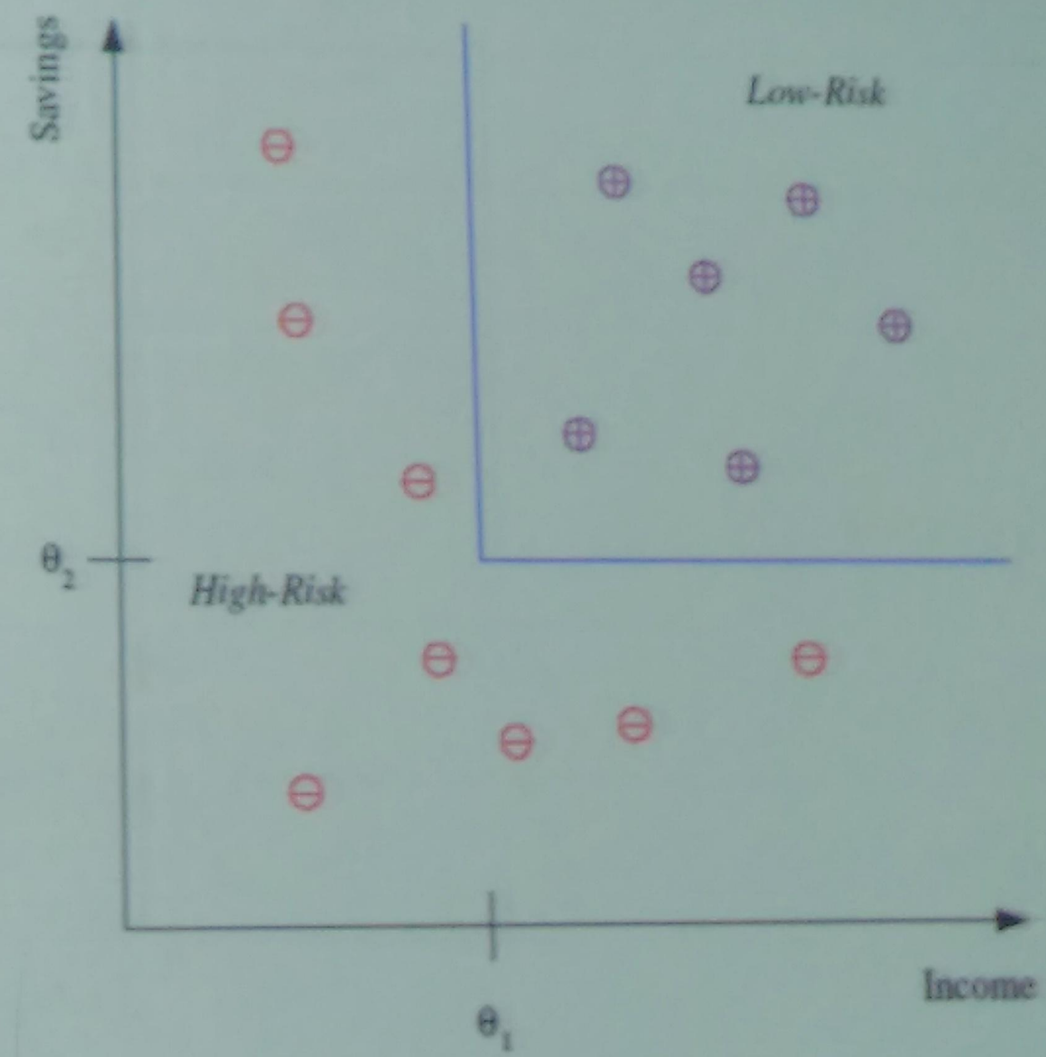
Testing: apply f to a never before seen *test example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

Major paradigms of machine learning

- **Rote learning** – One-to-one mapping from inputs to stored representation. "Learning by memorization." Association-based storage and retrieval.
- **Induction** – Use specific examples to reach general conclusions
- **Clustering** – Unsupervised identification of natural groups in data
- **Analogy** – Determine correspondence between two different representations
- **Discovery** – Unsupervised, specific goal not given
- **Genetic algorithms** – "Evolutionary" search techniques, based on an analogy to "survival of the fittest"
- **Reinforcement** – Feedback (positive or negative reward) given at the end of a sequence of steps

Classification

- Example: Credit scoring
- Differentiating between low-risk and high-risk customers from their *income* and *savings*

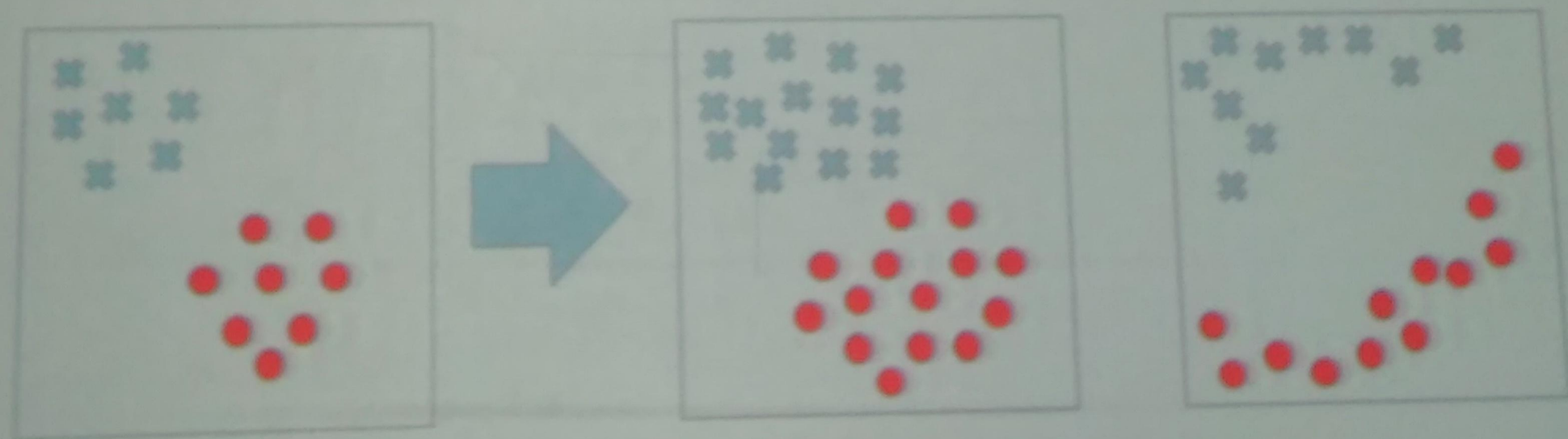


Discriminant: IF *income* > θ_1 AND *savings* > θ_2
THEN low-risk ELSE high-risk

Training and testing

- Training is the process of making the system able to learn.

- No free lunch rule:
  - Training set and testing set come from the same distribution
  - Need to make some assumptions or bias



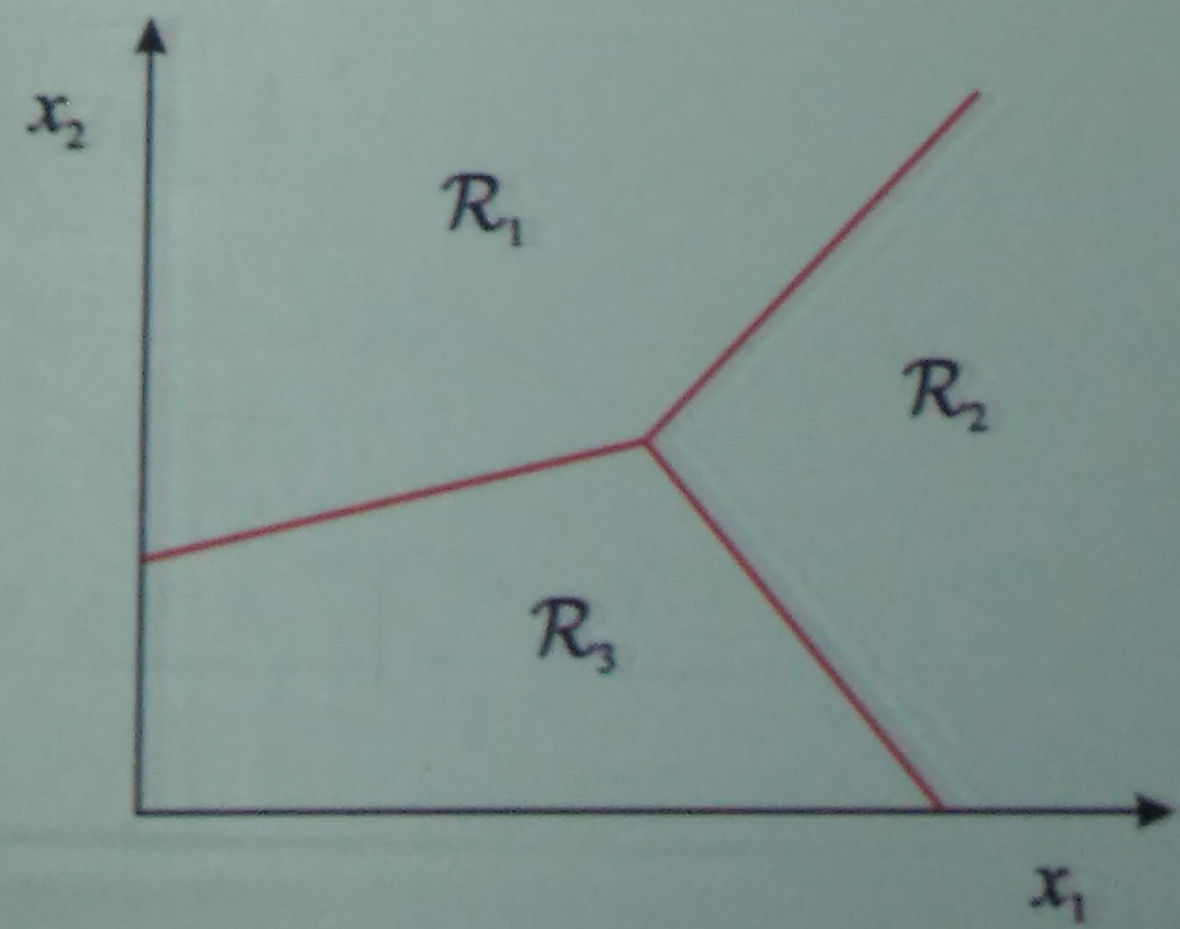
Performance

- There are several factors affecting the performance:
  - **Types of training** provided
  - The form and extent of any initial **background knowledge**
  - The **type of feedback** provided
  - The **learning algorithms** used

- Two important factors:
  - Modeling
  - Optimization

Classification

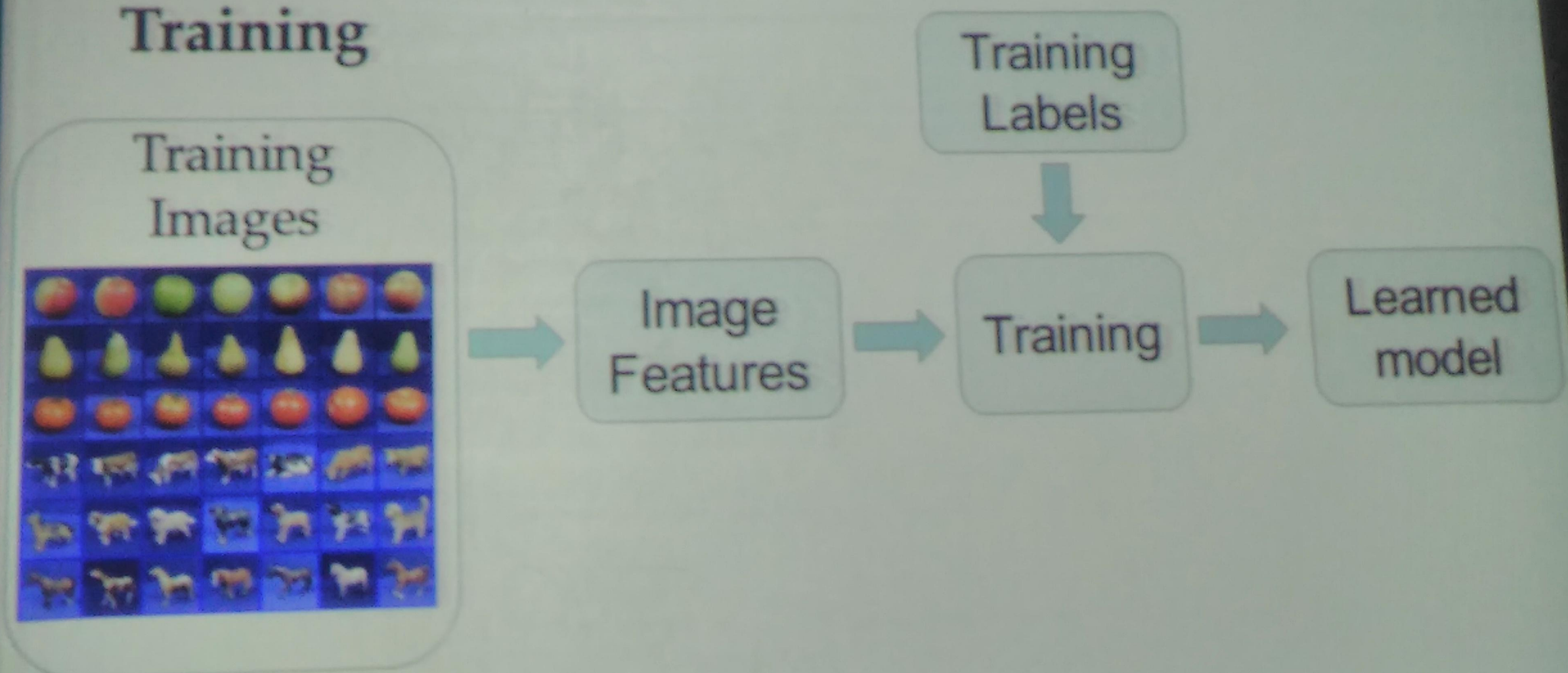
- Assign input vector to one of two or more classes
- Any decision rule divides input space into *decision regions* separated by *decision boundaries*



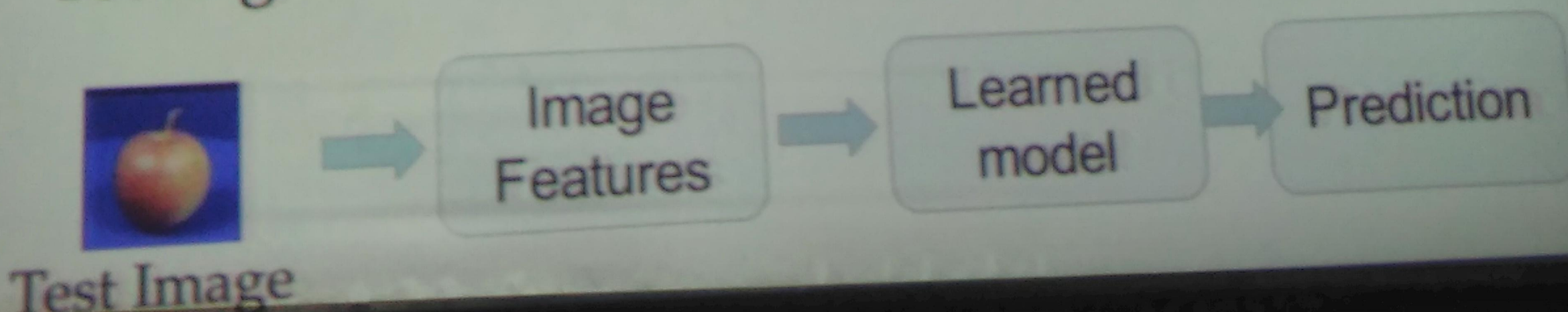
Slide credit: L. Lazebnik

Steps

Training



Testing



Prediction: Regression

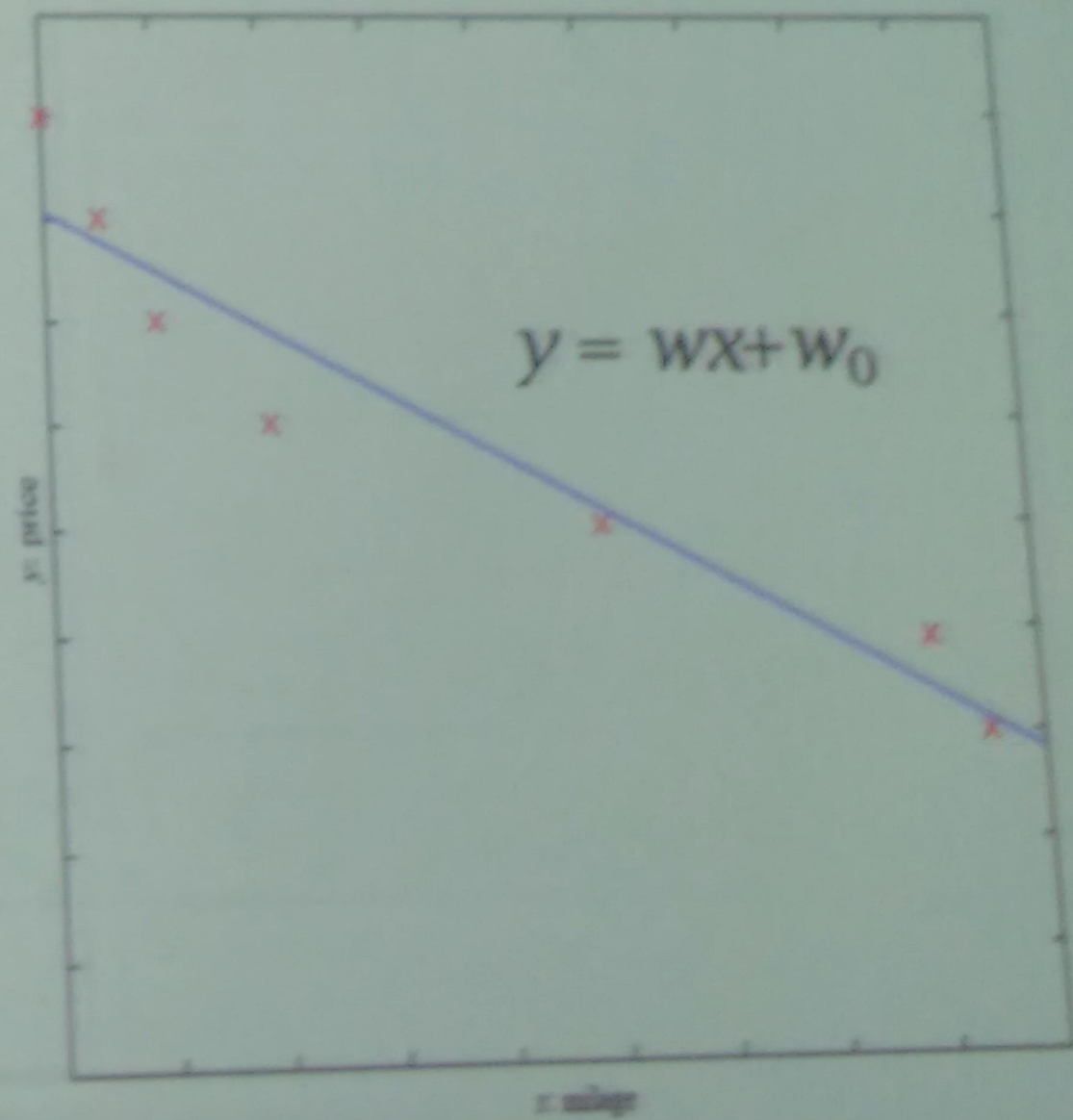
- Example: Price of a used car
- x : car attributes

  y : price

  $$y = g(x \mid \theta)$$

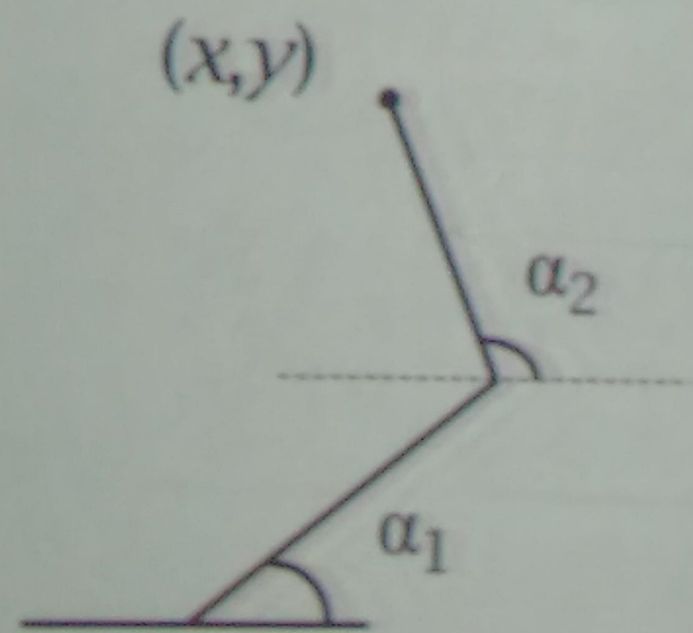
  $g(\,)$ model,

  θ parameters



Regression Applications

- Navigating a car: Angle of the steering wheel (CMU NavLab)
- Kinematics of a robot arm



$$\alpha_1 = g_1(x,y)$$

$$\alpha_2 = g_2(x,y)$$

Inductive Learning

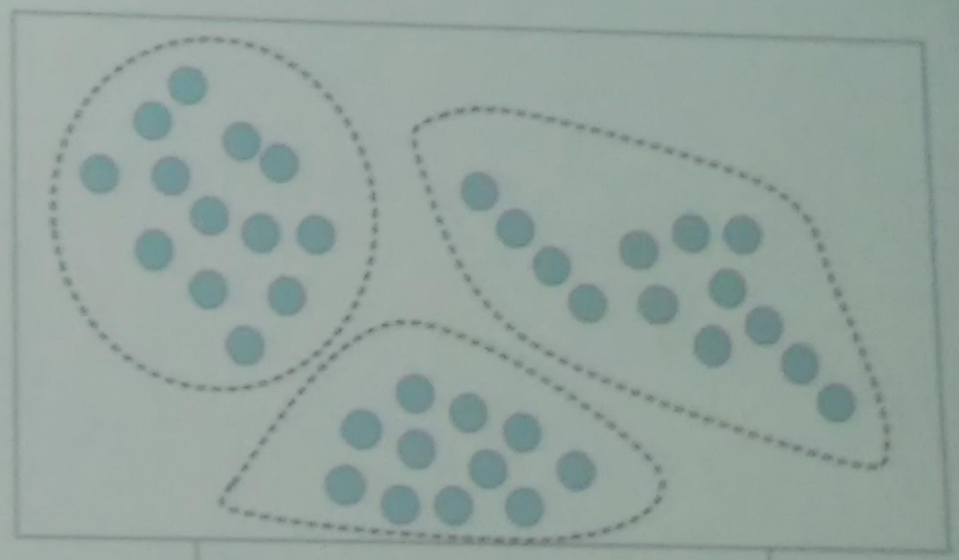
- **Given** examples of a function *(X, F(X))*
- **Predict** function *F(X)* for new examples *X*
  - Discrete *F(X)*: Classification
  - Continuous *F(X)*: Regression
  - *F(X)* = Probability(*X*): Probability estimation

Algorithms

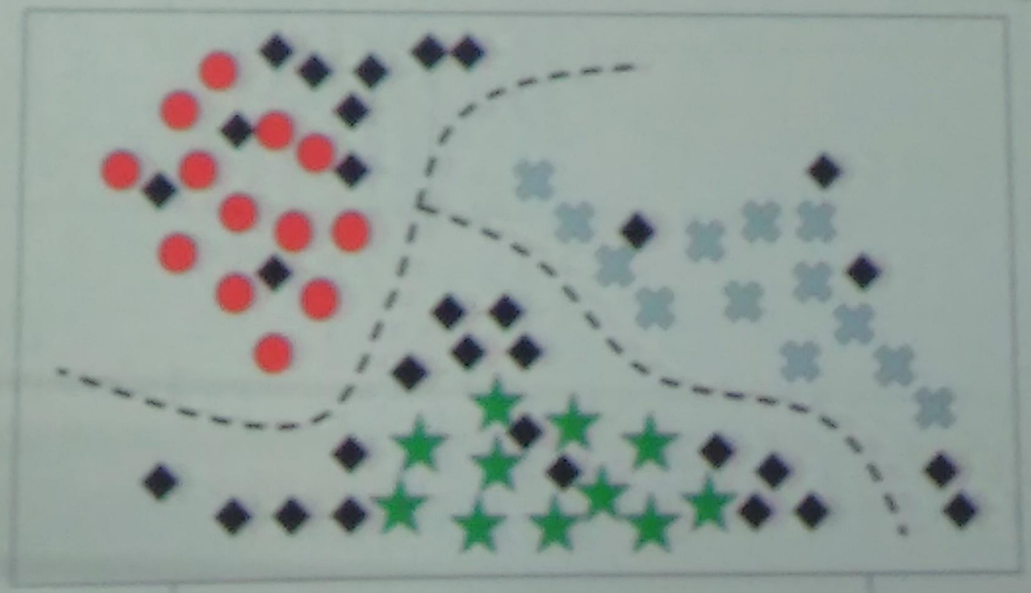
- The success of machine learning system also depends on the algorithms.
- The algorithms control the search to find and build the knowledge structures.
- The learning algorithms should extract useful information from training examples.

Algorithms

Supervised learning



Unsupervised learning



Semi-supervised learning

Algorithms

- **Supervised learning** ($\{x_n \in R^d, y_n \in R\}_{n=1}^N$)
  - Prediction
  - Classification (discrete labels), Regression (real values)
- **Unsupervised learning** ($\{x_n \in R^d\}_{n=1}^N$)
  - Clustering
  - Probability distribution estimation
  - Finding association (in features)
  - Dimension reduction
- **Semi-supervised learning**
- **Reinforcement learning**
  - Decision making (robot, chess machine)

What are we seeking?

- Supervised: Low E-out or maximize probabilistic terms

$$error = \frac{1}{N}\sum_{n=1}^{N}[y_n \neq g(x_n)]$$

E-in: for training set

E-out: for testing

$$Eout(g) \leq Ein(g) \pm O\left(\sqrt{\frac{d_{VC}}{N}\ln N}\right)$$

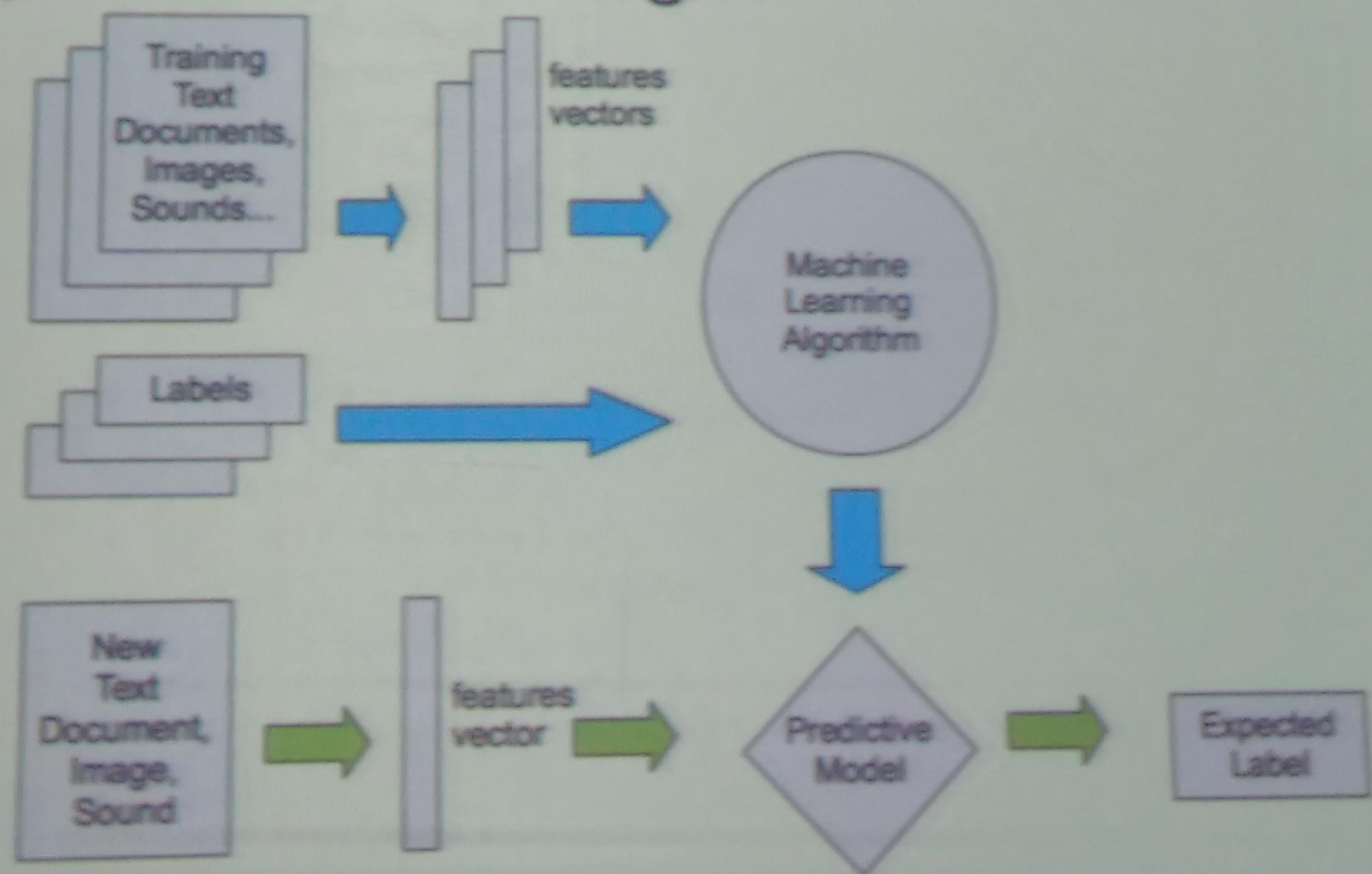
- Unsupervised: Minimum quantization error, Minimum distance, MAP, MLE(maximum likelihood estimation)

Unsupervised Learning

- Learning "what normally happens"
- No output
- Clustering: Grouping similar instances
- Other applications: Summarization, Association Analysis
- Example applications
  - Customer segmentation in CRM
  - Image compression: Color quantization
  - Bioinformatics: Learning motifs

Machine learning structure

- Supervised learning



Reinforcement Learning

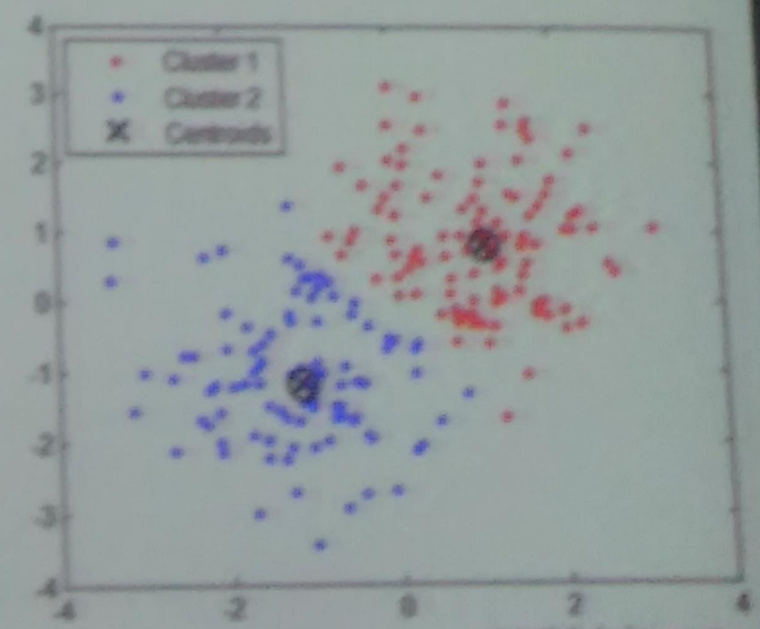
- Topics:
  - Policies: what actions should an agent take in a particular situation
  - Utility estimation: how good is a state (→used by policy)
- No supervised output but delayed reward
- Credit assignment problem (what was responsible for the outcome)
- Applications:
  - Game playing
  - Robot in a maze
  - Multiple agents, partial observability, ...

Clustering Analysis

- Definition

  Grouping unlabeled data into clusters, for the purpose of inference of hidden structures or inform.

- Dissimilarity measurement
  - Distance : Euclidean(L_2), Manhattan(L_1)
  - Angle : Inner product, ...
  - Non-metric : Rank, Intensity, ...
- Types of Clustering
  - Hierarchical
    - Agglomerative or divisive
  - Partitioning
    - K-means, VQ, MDS, ...



(Matlab helppage)

K-Means

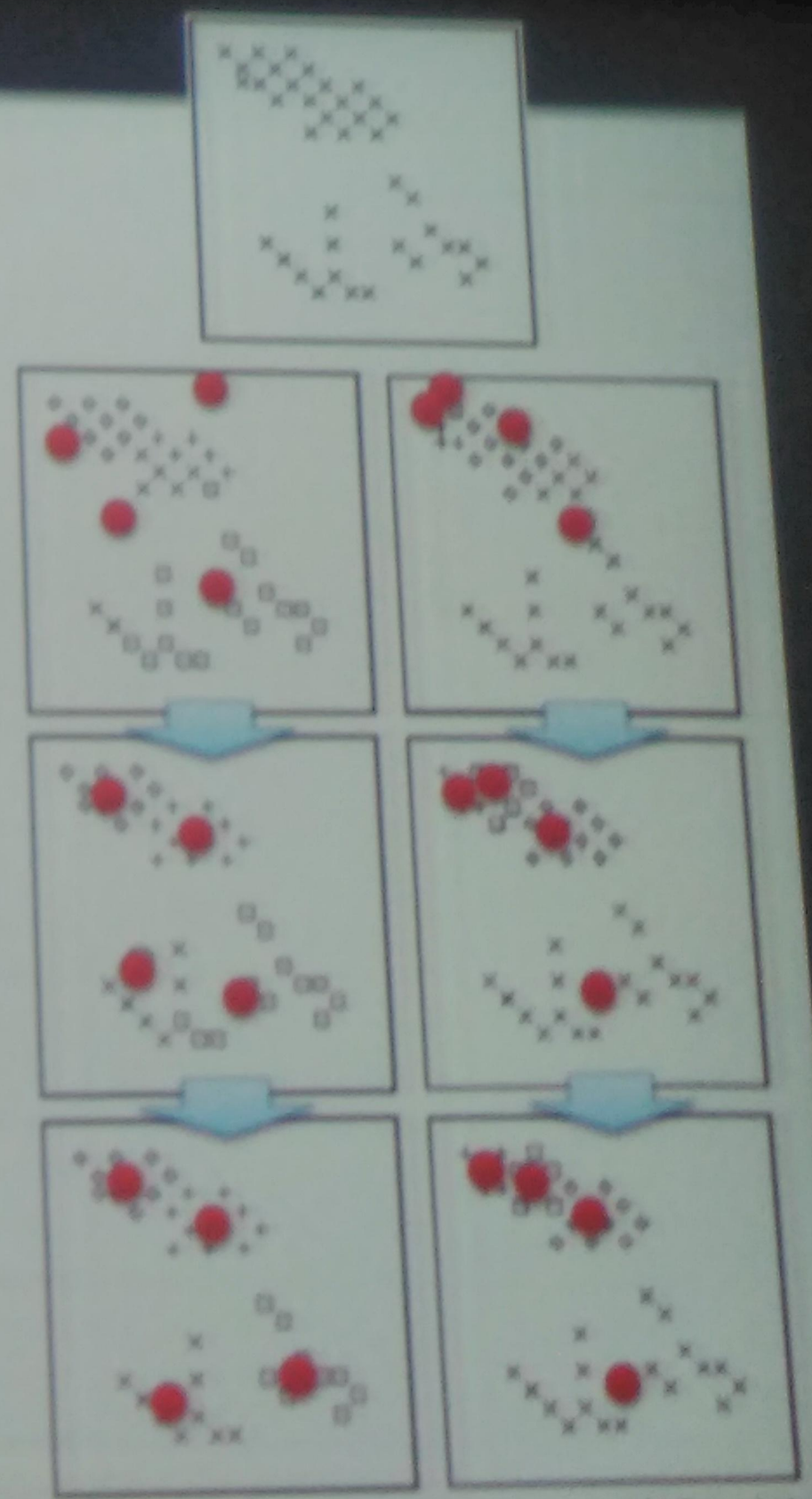
- Find K partitions with the total intra-cluster variance minimized

$$E = \sum_{i=1}^{K} \sum_{\mathbf{x} \in C_i} (\mathbf{x} - \mathbf{y}_i)^2$$

- Iterative method
  - Initialization : Randomized y_i
  - Assignment of x (y_i fixed)
  - Update of y_i (x fixed)

- Problem?

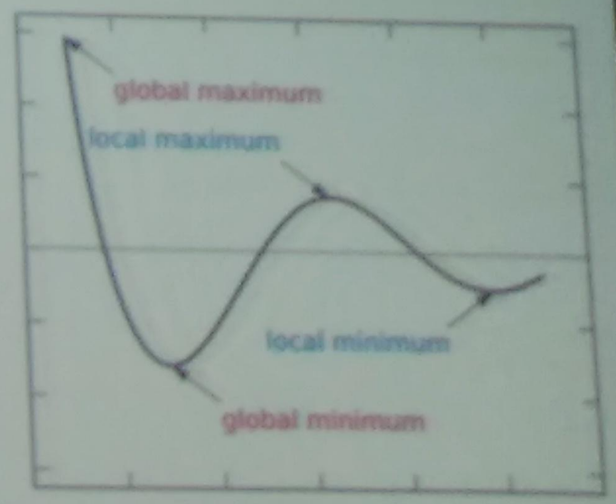
  → Trap in local minima



(MacKay, 2003)44

Deterministic Annealing (DA)

- Deterministically avoid local minima
  - No stochastic process (random walk)
  - Tracing the global solution by changing level of randomness



(Maxima and Minima, Wikipedia)

- Statistical Mechanics
  - Gibbs distribution

$$P(E_x) = \exp(-E_x/T)/Z_x \qquad Z_x = \sum_{x \in \Omega} \exp(-E_x/T)$$

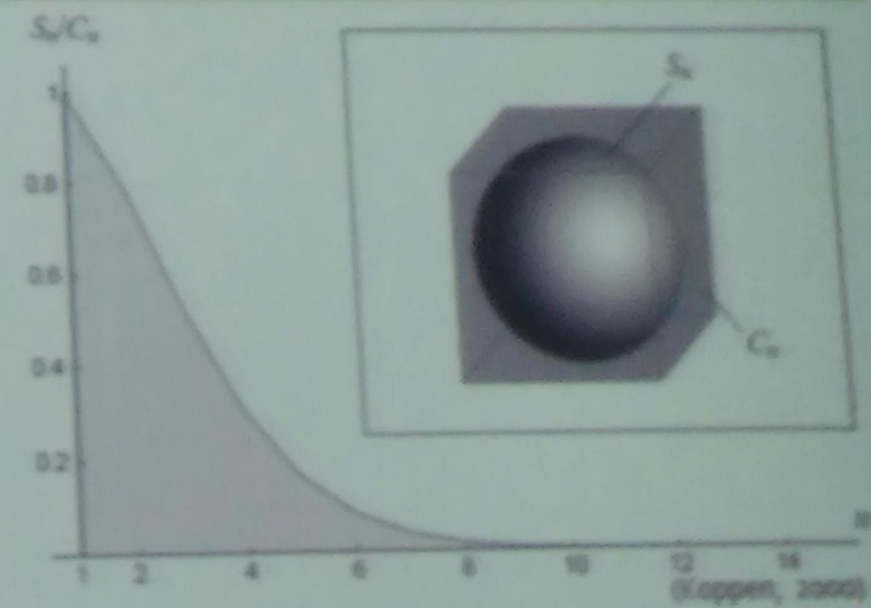
  - Helmholtz free energy F = D – TS
    - Average Energy $D = \langle \sum E_x \rangle$
    - Entropy $S = -P(E_x) \ln P(E_x)$
    - $F = -T \ln Z$
- In DA, we make F minimized

Dimension Reduction

- Definition

  Process to transform high-dimensional data into low-dimensional ones for improving accuracy, understanding, or removing noises.

- Curse of dimensionality
  - Complexity grows exponentially in volume by adding extra dimensions



- Types
  - Feature selection : Choose representatives (e.g., filter,...)
  - Feature extraction : Map to lower dim. (e.g., PCA, MDS, ...)

Naïve Bayes Example

- Let's understand it using an example.
  - Have a training data set of weather and corresponding target variable 'Play'.
  - Now, we need to classify whether players will play or not based on weather condition.
  - Let's follow the below steps to perform it.
    - Step 1: Convert the data set to frequency table
    - Step 2: Create Likelihood table by finding the probabilities like
      - Overcast probability = 0.29 and
      - probability of playing is 0.64.

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Naïve Bayes

- Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class.
  - The class with the highest posterior probability is the outcome of prediction.

- **Problem:**
  - Players will pay if weather is sunny, is this statement is correct?
  - We can solve it using above discussed method,
    - so P(Yes | Sunny) = P(Sunny | Yes) * P(Yes) / P (Sunny)
    - Here we have, P (Sunny |Yes) = 3/9 = 0.33,
      P(Sunny) = 5/14 = 0.36,
      P(Yes)= 9/14 = 0.64
    - Now, P (Yes | Sunny) = 0.33 * 0.64 / 0.36 = 0.60,
    - which has higher probability.

- Naive Bayes uses a similar method to
  - predict the probability of different class based on various attributes.
  - This algorithm is mostly used in text classification and
  - with problems having multiple classes.

Naïve Bayes

- Bayes theorem provides a way of calculate
  - posterior probability P(c|x) from P(c), P(x) and P(x|c).
- Look at the equation belov

Likelihood

Class Prior Probability

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Posterior Probability

Predictor Prior Probability

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \cdots \times P(x_n|c) \times P(c)$$

  - Here,
  - *P*(*c*|*x*) is the posterior probability of *class* (*target*) given *predictor* (*attribute*).
  - *P*(*c*) is the prior probability of *class*.
  - *P*(*x*|*c*) is the likelihood which is the probability of *predictor* given *class*.
  - *P*(*x*) is the prior probability of *predictor*

Naïve Bayes Example

- Let's understand it using an example.
  - Have a training data set of weather and corresponding target variable 'Play'.
  - Now, we need to classify whether players will play or not based on weather condition.
  - Let's follow the below steps to perform it.
    - Step 1: Convert the data set to frequency table
    - Step 2: Create Likelihood table by finding the probabilities like
      - Overcast probability = 0.29 and
      - probability of playing is 0.64.

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Introduction

- Brain MRI images play an important role to diagnosis brain tumor.
- White matter (WM), gray matter (GM) and cerebro-spinal fluid (CSF) is eminent characteristics for the better understanding of structural and functional activities of human brain for scientific research and clinical diagnosis
- The brain MRI Image segmentation process is complex due to the noise, heterogeneity and bias field present within the brain MRI image.

Motivation

- Noise and intensity inhomogeneity caused by bias field i.e. magnetic settings, patients' position, and other unusual factors.
- A new local spatial membership function inherent within the unsupervised modified spatial fuzzy C-mean method to compute the optimized cluster centers.

The focus of the method is

- Maximum clustering accuracy in terms of partial coefficient and entropy.
- Minimum number of iteration and reduce the computational time

Preliminaries

- **Fuzzy C-means[10]:**
  - Fuzzy c-means algorithm efficiently segments the MRI images
  - Let, N no. of element to be segmented with C no. of optimum cluster
  - The only requirement to measure the optimum cluster centers, (V_i , *where* $i=1,2,\ldots C$)
  - The image is segmented depending on the membership function of each element with respect to the clusters

Preliminaries

- The iteration is stopped only when the cluster centers are optimized using the objective function (J_{fcm})

$$J_{fcm} = \sum_{i=1}^{C} \sum_{j=1}^{N} \mu_{ij}^{m} d_{ij}^{2}$$

- The Euclidian distance, d_{ik} is the intensity difference between the kth element and i^{th} cluster center. m is a fuzzifier ($m>=1$), used to set the weight of the objective function

$$d_{ik}^{2} = |x_k - V_i|^2$$

Novel Approach for Brain Image Segmentation

- The basic segments of human brain are White matter(WM), Gray matter(GM) and Cerebro-Spinal Fluid(CSF) etc.
- The complexity of the brain image segmentation is higher due to noise and intensity inhomogeneity caused by bias field i.e. magnetic settings, patients' position, and other unusual factors.
- Correlation among the neighbors of pixels of brain MRI image is higher
- The probability of belonging to the same cluster is higher for the highly correlated pixels.

Preliminaries

- **Partial Entropy (V_{pe}):**
  - Minimum value implies relation of pixels in same cluster is higher and different cluster is lower and vice versa

$$V_{pe} = \frac{\sum_{i=1}^{C}\sum_{j=1}^{N} U_{ij} * \log(U_{ij})}{N}$$

  - Where, U_{ij} is the membership function

Preliminaries

- The local membership function (u_{ik}) is defined as

$$u_{ij} = \frac{(f_{ij})^{\frac{1}{m-1}}}{\sum_{k=1}^{c}\left(\frac{d_{ij}^{2}}{d_{kj}^{2}}\right)^{\frac{2}{(m-1)}}}$$

- The local and global (same as FCM, μ_{ij}) membership is combined (as Z_{ij}) to get the probability (fuzziness) of the pixels with respect to the different cluster centers

$$Z_{ij} = \frac{\mu_{ij}^{p} * u_{ij}^{q}}{\sum_{k}^{c} \mu_{kj}^{p} * u_{kj}^{q}}$$

- Contribution of the membership functions are controlled using two variable p and q

- ***After each iteration new cluster centers are calculated as,***

$$W_i = \frac{\sum_{j=1}^{N} Z_{ij}^{m} x_j}{\sum_{j=1}^{N} Z_{ij}^{m}}$$

Preliminaries

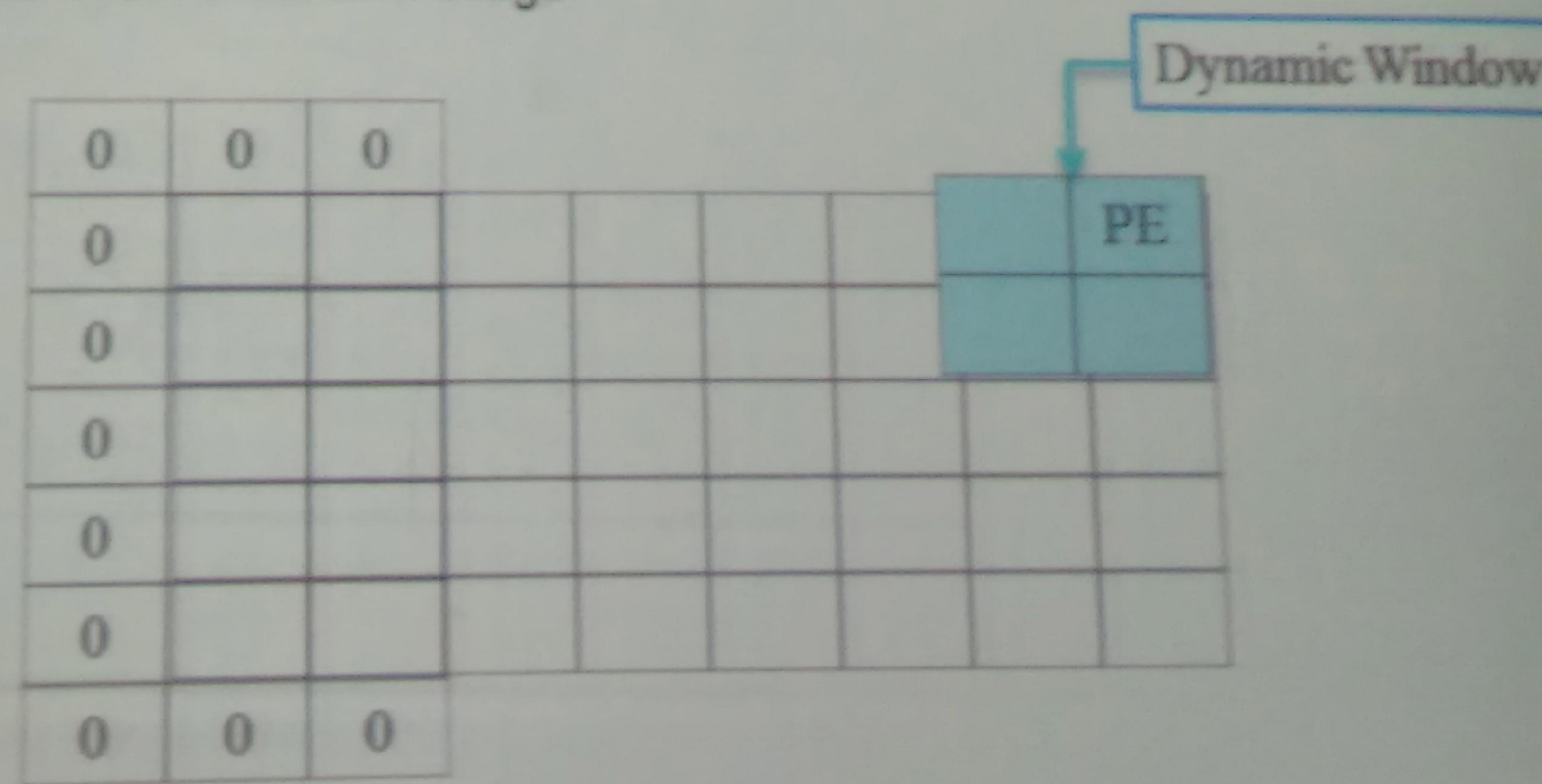
- **Partial Entropy (V_{pe}):**
  - Minimum value implies relation of pixels in same cluster is higher and different cluster is lower and vice versa

$$V_{pe} = \frac{\sum_{i=1}^{C}\sum_{j=1}^{N} U_{ij} * \log(U_{ij})}{N}$$

- Where, U_{ij} is the membership function

Novel Approach for Brain Image Segmentation

- Segmentation method generate optimum cluster centers with minimum iteration
- Designed based on dynamic windowing (Masking) i.e changes its size: 2x2, 2x3 or 3x2 and 3x3 at corner, border and rest of the image respectively to avoid padding.
- The number of cluster centers (**C**) and their initial values (V_i) are taken from the histogram analysis of the brain MRI image

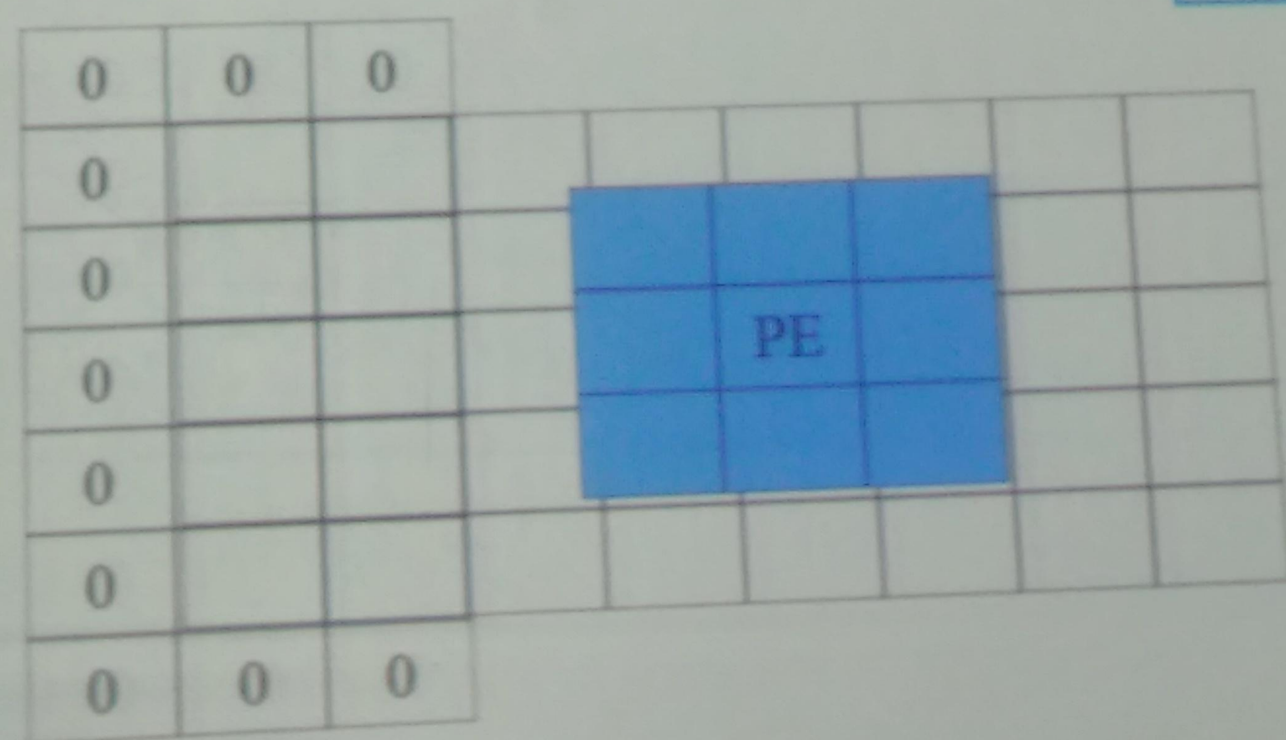


Novel Approach for Brain Image Segmentation

- Segmentation method generate optimum cluster centers with minimum iteration
- Designed based on dynamic windowing (Masking) i.e changes its size: 2x2, 2x3 or 3x2 and 3x3 at corner, border and rest of the image respectively to avoid padding.
- The number of cluster centers (***C***) and their initial values (V_i) are taken from the histogram analysis of the brain MRI image

Dynamic Window

Static Window



Novel Approach for Brain Image Segmentation

- The proposed method iteratively optimized the cluster centers using global membership value (u_{ij}) and local spatial information.

- Iteration is terminated when number of clusters centers are optimal.

- Optimal cluster centers are obtained when the absolute differences between the intensity of old and new cluster centers are less than the predefined threshold ($Th=0.00005$).

Novel Approach for Brain Image Segmentation

- The local spatial information is used to measure the correlation among the neighboring pixels
- To measure the correlation we proposed a Local Spatial Membership (L_{ij}) is the probability of x_j pixel to belong in i^{th} cluster

$$L_{ij} = \frac{\sum_{k=1}^{M} d_{ik}^2 | d_{ik} = \min(d_{ik}) \in V_i \forall i,k}{\sum_{i=1}^{C} \sum_{k=1}^{M} d_{ik}^2 | d_{ik} = \min(d_{ik}) \in V_i \forall i,k}$$

C and M are the number of cluster and size of the dynamic window respectively

Novel Approach for Brain Image Segmentation

- To understand the local spatial membership function, consider the window (mask) M_j for j^{th} voxel (or x_j as processing element).

- The size of the window is M and Euclidian distance between x_k and V_i as

$d_{ik}^2 = |x_k - V_i|^2$ Where x_k is the kth element in the mask, M_j *and* $k=1,2,..,M$

Window, M_j

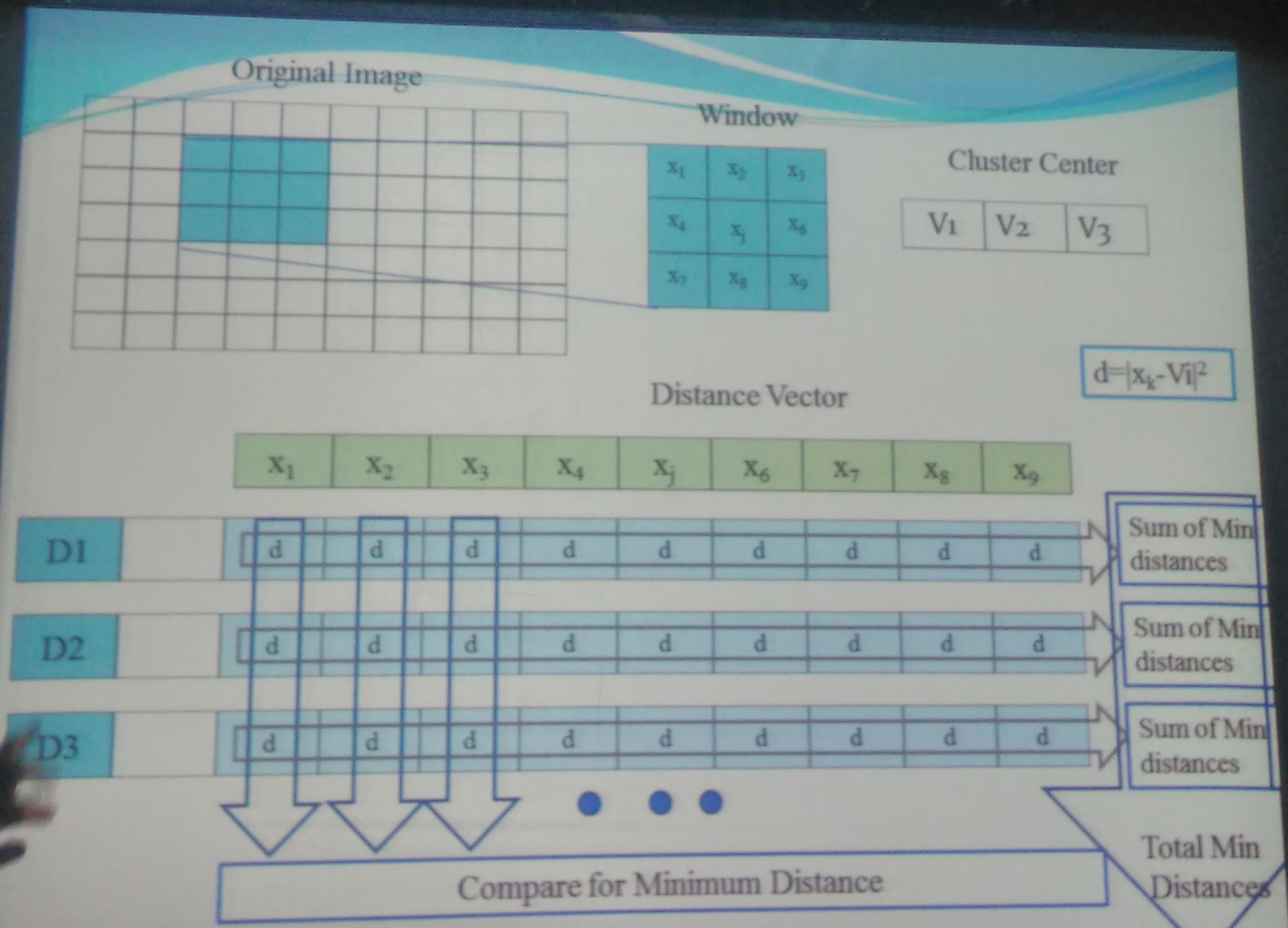
x_1	x_2	x_3
x_4	x_j	x_6
x_7	x_8	x_9

Novel Approach for Brain Image Segmentation

- Depending on the number of cluster centers, we get C number of **distance vector**, $D_i[k]$ *i.e.* $d_{ik}^2 \in D_i[k]$

- Comparing among the kth element of all the distance vector, $D_i[k]$, we get the minimum distance between x_k and V_i and the number of minimum distance for each distance vector, $D_i[k]$, for all k=1, 2, …, M.

- Measure the sum of the minimum differences for i^{th} distance vector, $D_i[k]$ *and* the sum of minimum differences for all distance vectors


Original Image
Window
x_1 x_2 x_3
x_4 x_5 x_6
x_7 x_8 x_9
Cluster Center
V1 V2 V3
$d=|x_k-V_i|^2$
Distance Vector
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9
D1
D2
D3
d
Sum of Min distances
Sum of Min distances
Sum of Min distances
Total Min Distances
Compare for Minimum Distance

Novel Approach for Brain Image Segmentation

- ***Local Membership Function*** (L_{ij}) is the ratio of the sum of the minimum differences for i^{th} distance vector ***and*** the sum of minimum differences for all distance vectors

- The local spatial membership values are used to group the neighbourhood voxels and suppress the intensity inhomogeneity and noise

Novel Approach for Brain Image Segmentation

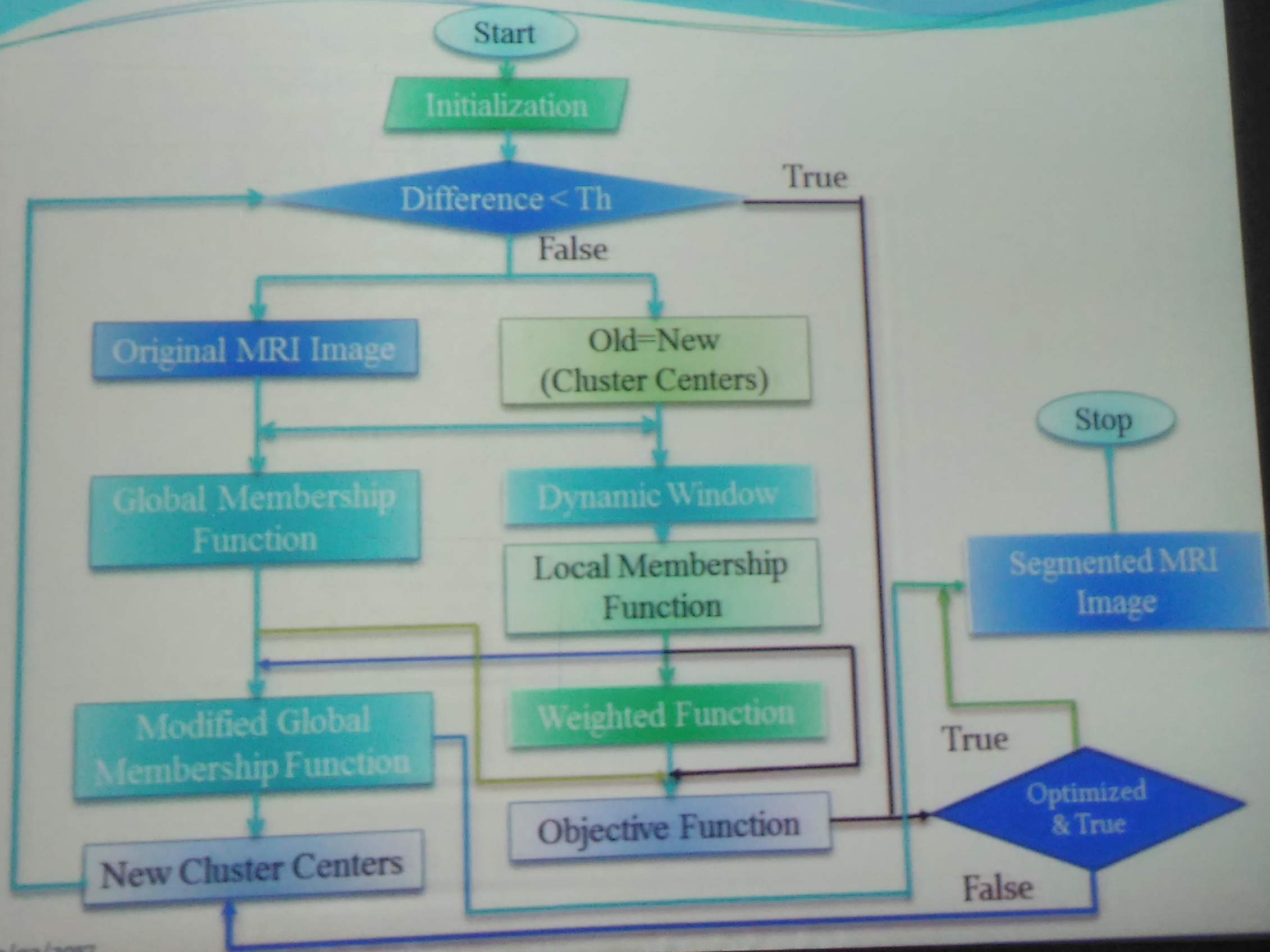
- Combining the global (same as FCM, μ_{ij}) and proposed local membership (L_{ij}) function
- we get a modified global membership (U_{ij}), defined as

$$U_{ij} = \frac{\mu_{ij}^{p} * L_{ij}^{q}}{\sum_{k=1}^{C} \mu_{kj}^{p} * L_{kj}^{q}}$$

p and q two user variable decide the contribution of the global and local information in modified global membership

- We set p=q=2 for better segmentation and as well as noise removal

Flowchart


Start
Initialization
Difference < Th
True
False
Original MRI Image
Old=New
(Cluster Centers)
Global Membership
Function
Dynamic Window
Local Membership
Function
Stop
Segmented MRI
Image
Modified Global
Membership Function
Weighted Function
True
Optimized
& True
Objective Function
New Cluster Centers
False

Result and Discussion

- ***Input:*** T1 weighted brain MRI image is taken from the freely available Brain Web dataset [26]

- Different combination of noise and intensity inhomogeneity(IIH) (percentage) for T1 weighted simulated brain MRI image have been chosen as,

Sl No.	*Noise*	*IIH*
1st	9%	40%
2nd	9%	20%
3rd	7%	40%
4th	7%	20%

- The image resolution of input brain MRI is 181x217x181 & size 1x1x1 mm

- **Simulation:** Matlab (v. 2012a), windows 7 with 4GB RAM and 3.10 GHz Intel(R) i3 processor

- The experimental results are compared with the recently published works

Result and Discussion

- The qualitative results of the proposed method provides White Matter (WM), Gray Matter (GM) and cerebro-spinal fluid (CSF), shown in figure 2
- The cluster validation functions, partial coefficient and partial entropy are used to measure the quantitative result
- The comparative quantitative results are shown in Table I
- Table I shows that the proposed method is 2% and 42% improved the partial coefficient and partial entropy respectively over recently published work SpFCM [3]

Table I

Segmented Image	Segmentation Technique	Validation Function	
		V_{pc}	V_{pe}
Noise 9% IIH 40%	FCM[10]	0.803	0.378
	sFCM[15]	0.897	0.180
	SpFCM[3]	0.968	0.052
	Proposed Method	**0.982**	**0.030**

Novel Approach for Brain Image Segmentation

- Depending on the number of cluster centers, we get C number of **distance vector**, $D_i[k]$ *i.e.* $d_{ik}^2 \in D_i[k]$

- Comparing among the kth element of all the distance vector, $D_i[k]$, we get the minimum distance between x_k and V_i and the number of minimum distance for each distance vector, $D_i[k]$, for all k=1, 2, ..., M.

- Measure the sum of the minimum differences for i^{th} distance vector, $D_i[k]$ ***and*** the sum of minimum differences for all distance vectors