Some Applications of MATLAB

Himanshu Mittal and Raju Pal

Department of Computer Science
Jaypee Institute of Information Technology, Noida

March 11-12, 2018

Outline

Outline

Introduction

- Analyze data from different perspectives and summarizing it into useful information.
- Discovers hidden patterns in databases.
- Integrates methods from several fields including statistics, database systems, machine learning, and pattern recognition, and is used to mine important and useful information by processing large volumes of data.

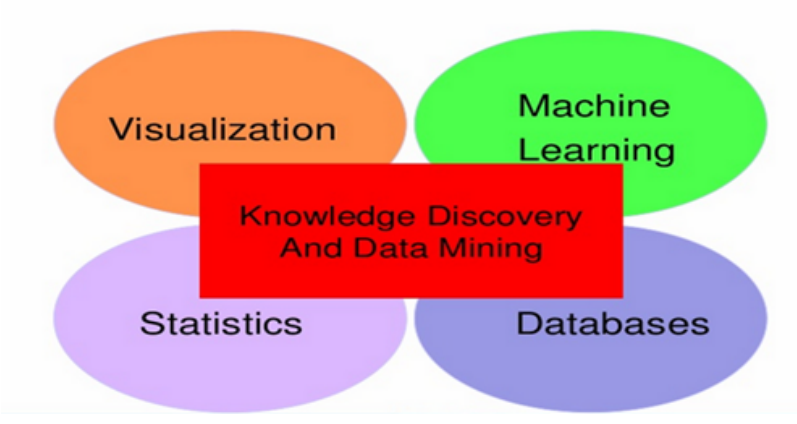


Figure: Knowledge discovery and Data mining

Data Mining Tasks

- Predictive tasks : predict the value of a particular attribute based on some past learning.
  - Classification
  - Regression
- Descriptive tasks : derive patterns that summarize the essential relationship in data.
  - Clustering
  - Association rule mining

Outline

Introduction

- This is an statistical data mining approach that find the relationship between the variables.
- It includes techniques for modeling and analyzing several variables:
  - Independent (or predictor)
  - Dependent (or outcome)
- Understands how the typical value of the dependent variable changes when any one of the independent variables is varied, while the other independent variables are held fixed.

Linear Regression

- Linear regression models the linear relationship between *predictor* and *response* variables.
- Linear relation is depicted as:

$$y = a_0 + a_1 * x + e$$

  where a_0 is the y-intercept, a_1 is the slope (or regression coefficient), and e is the error term.
- X as predictor variable and Y as outcome variable.
- *least-square fit* is commonly used to fit the linear regression.
- Perform *correlation analysis* to establish if a linear relationship exists between these quantities.

Linear Regression

- Eg; predict *accidents* in unknown state using its *Total population* using *linear regression* (only one independent variable in the relation)
- Using *accidents* dataset with two features *Total population of state* and *accidents per state*
  - From the dataset *accidents*
    load accidents;
  - Load accident data in y and state population data in x.
    x = hwydata(:,14); %Population of states
    y = hwydata(:,4); %Accidents per state
  - Perform correlation analysis:
    corrcoef([x y])
    As, all correlation coefficients are close to 1, means *strong positive correlation* between each pair of data columns
  - Find the linear regression relation of form: $y = a_0 + a_1 * x$
  - Include y-intercept a_0 by padding 'x' with a column of ones
    X = [ones(length(x),1) x];
  - Using the \ operator (\ performs a least-squares regression)
    b = X\y;
  - The linear relation should: (use ***format long*** to the values)
    *y=142.7120 + 0.0001256*x*

Linear Regression

- Calculate the *accidents per state 'yCalc'* from 'x' using the formed relation:

  *yCalc1 = X*b;*

- Visualizing the regression by plotting actual values *'y'* and the calculated values *'yCalc'*:

  scatter(x,y);
  hold on
  plot(x,yCalc1)
  xlabel('Population of state')
  ylabel('Fatal traffic accidents per state')
  title('Linear Regression Relation Between Accidents & Population')

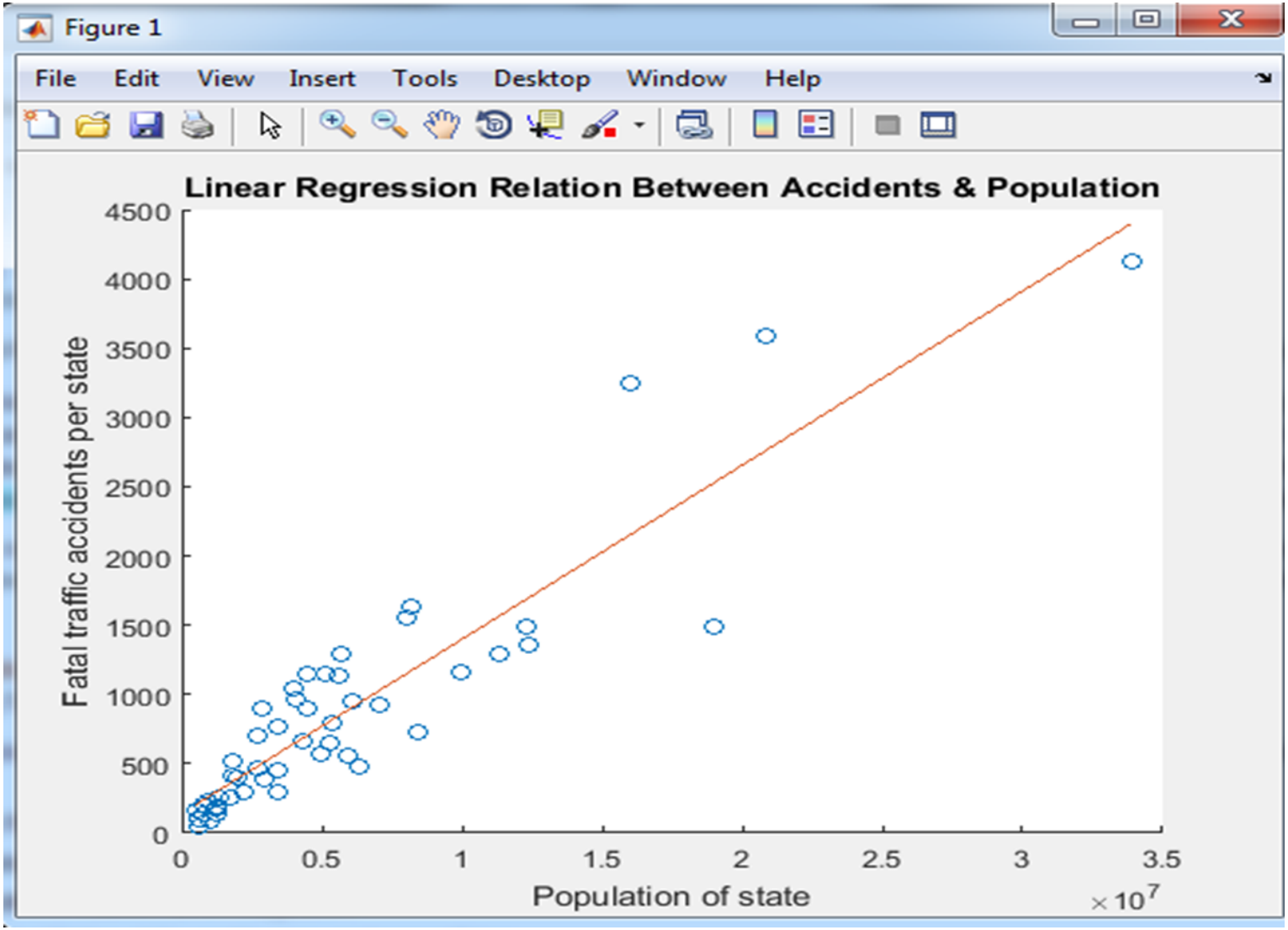
- *The predicted value of accidents 'ypred' for x_1 = 800000*

  X_1 = [ones(length(x1),1) x1];
  *ypred = X_1*b*

- predicted value: (use ***format short*** to the values)

  ypred = 243.22

Linear Regression

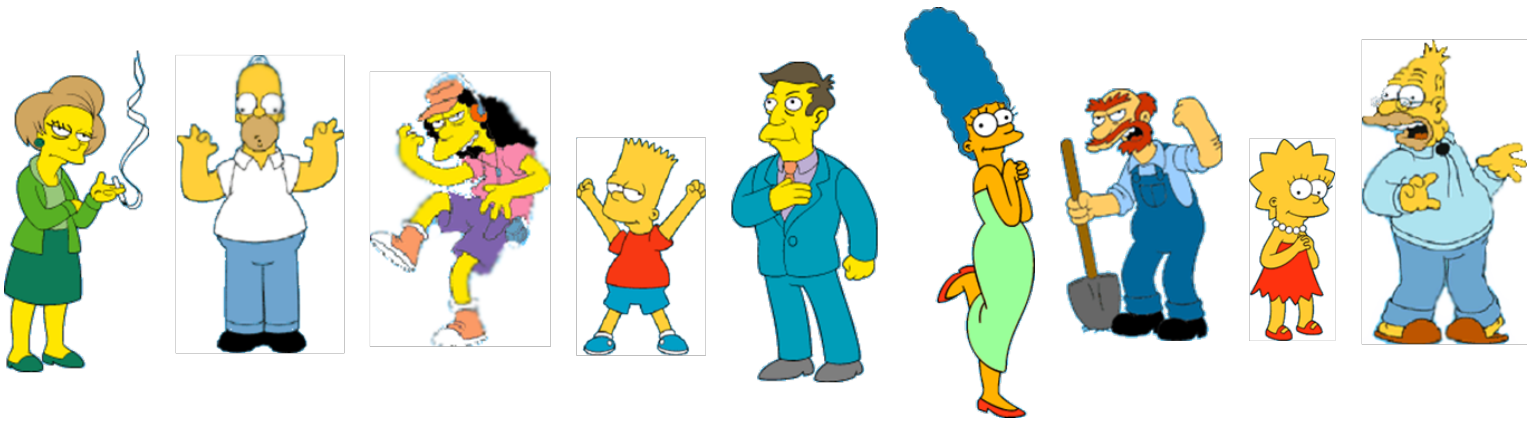


Outline

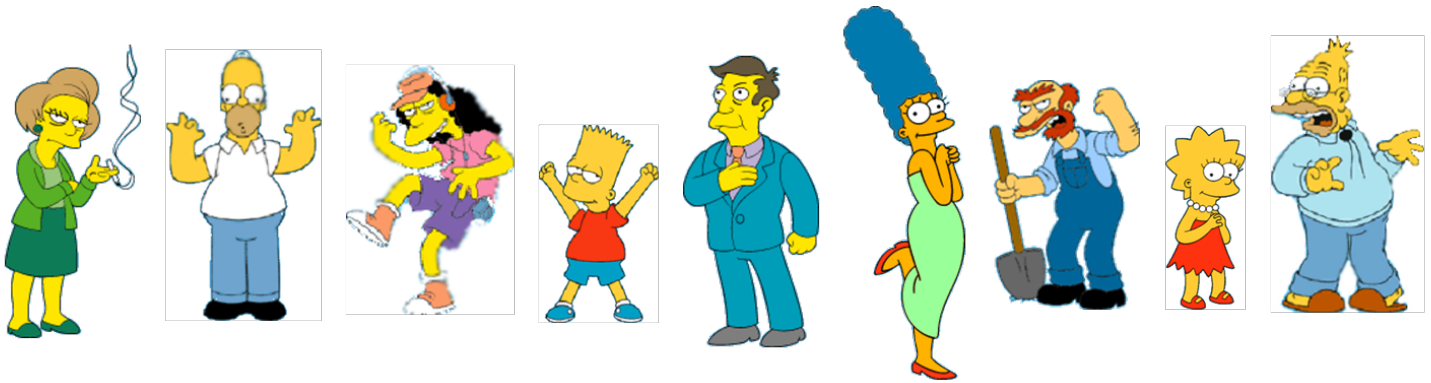
What is Clustering?

- Organizing data into groups or classes such that there is
  - high intra-class similarity
  - low inter-class similarity
- Finding the class labels and the number of classes directly from the data
  (in contrast to classification)
- More informally, finding natural groupings among objects.

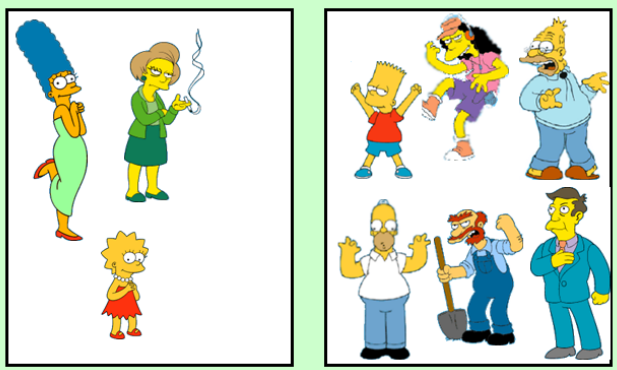
What is a natural grouping among these objects?



What is a natural grouping among these objects?



Clustering is subjective



Females Males

K-Means

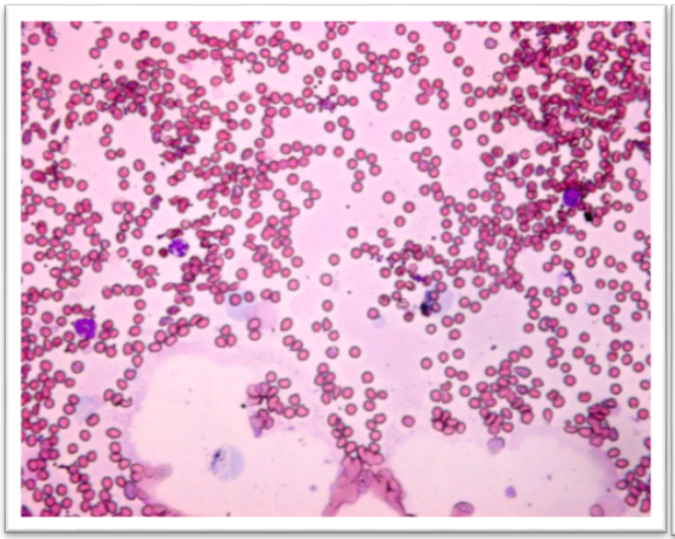
- Widely used algorithm for performing clustering
- k-means minimizes within-cluster point scatter
- Dissimilarity measure: Euclidean distance metric
- **Algorithm:**
  1. Randomly choose ***k*** data items from dataset X as initial centroids.
  2. Repeat until the convergence criteria is met.
     - Assign each data point to the cluster with closet centroid based on the distance measure.
     - Update cluster centroids using the mean of data items.
- Number of clusters to be created needs to be known inadvance
- Different initial partitions result in different final clusters

Clustering using k-Means

- Eg; identify the cluster for newdata (1.2,0.5) using the 'fisher's iris' dataset.
  - Load 'Fisher's iris' data set.
    load fisheriris;
  - Use the petal lengths and widths as predictors.
    x = meas(:,3:4);
  - Clustering the data with k = 3 clusters
    [idx,C] = kmeans(x,3);
    where, idx is a column vector of predicted cluster indices for each observation and C is matrix of centroid locations.
  - Load the data of cluster to be identified
    newData=[1.2 0.5]
  - Calculate the distance b/w newData and identified centroids
    dis=pdist2(C,newData)
  - Find the cluster with minimum distance
    [M,I] = min(dis)
    where, *M* represents the distance of newData from identified cluster and *I* represents the cluster number

What is an Image?

- An image is a visual representation of some measurable property of person, object, or phenomenon

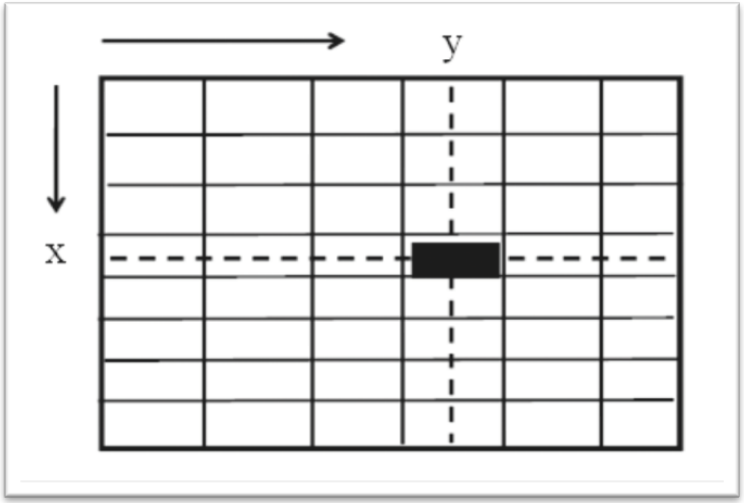


Blood smear image

Nature image

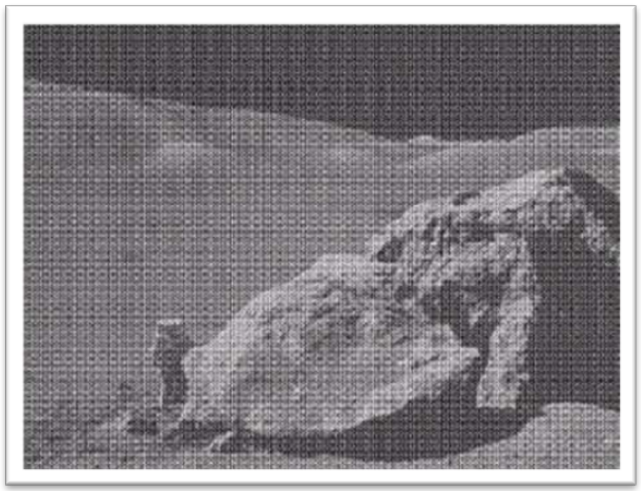
What is a Digital Image?

- In mathematical notation, digital image is defined as a 2-D function, f(x,y), where x and y denote spatial coordinates and f is called intensity or gray level at the point (x,y)



What is a Digital Image Processing?

- Image processing is a data processing procedure of image using computer in order to perform a special task



Satellite image having noise

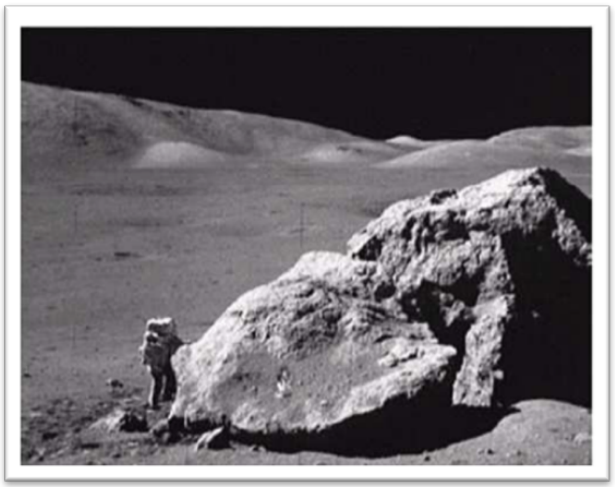


Image after noise removal

Principal Application Areas

- Improvement of pictorial information for human interpretation

- Processing of image data for storage, transmission and representation for machine perception

Application Areas

- Optical character recognition
- Medical applications
- Biometrics
- Weather forecasting
- Aging problem
- Other areas like facial expressions, etc.

Outline

Read/Write Image

- So we have an image file how do I access the info?
- Open up MATLAB and change working directory to where image is stored
- Use the imread() function
  im = imread('name_of_image.ext')
  - Use single quotes, and type in the full name of the image with its extension (.bmp, .jpg, etc.)
  - im will contain a 2D matrix (rows x cols) of B&W and Grayscale values or a 3D matrix (rows x cols x 3) of colour values
  - Matrix corresponds to each pixel in the digital image

Read/Write Image

- How do I access a pixel in MATLAB --→ B&W or Grayscale case?
  - pix = im(row,col);
    - row & col: Row & column of the pixel to access
    - pix contains the intensity value
    - Access elements in an array by round braces, not square!
  - For you C buffs Indexing starts at 1, not 0!
- How do I access a pixel in MATLAB --→ Colour case?
  - pix = im(row,col,1); --→ Red colour value
  - pix = im(row,col,2); --→ Green colour value
  - pix = im(row,col,3); --→ Blue colour value
  - 3rd argument --→ 3rd dimension of matrix
  - Only grabs one colour value at a time!

Read/Write Image

- How can I get the RGB pixel entirely? Use the : command
  pix = im(row,col,:);
  - : means to grab all values of one dimension
  - However, this will give you a 1 x 1 x 3 matrix we just want an array! Call the squeeze() command
    pix = squeeze(im(row,col,:));
  - Now a 3 x 1 vector. To access R, G and B values, do:
    red = pix(1) --→ Red,
    gr = pix(2) --→ Green,
    blue = pix(3) --→ Blue

Read/Write Image

- So I know how to get pixels; how can I modify them in the image?
  - Easy! Just go backwards
  - For a B & W Image do:
    im(row,col) = pix;
  - For a colour image, do either:
    im(row,col,1) = red;
    im(row,col,2) = green;
    im(row,col,3) = blue; or
    im(row,col,:) = [red; green; blue] or
    im(row,col,:) = rgb; %rgb - 3 x 1 vector

Read/Write Image

- How do I access a subset of the image?
  - How do I grab a portion of the image and store it into another variable?
- Do the following for monochromatic images:
  im2 = im(row1:row2,col1:col2);
- Do the following for colour images:
  im2 = im(row1:row2,col1:col2,:);
- This will grab a rectangular region between rows 1 and 2, and columns 1 and 2
- e.g., if I wanted to get rows $17 - 31$, and columns $32 - 45$ for colour, do:
  im2 = im(17:31,32:45,:);

Read/Write Image

- So I know how to get pixels; how can I display images?
- Use the imshow() command
  - imshow(im); --→ im Image loaded into MATLAB
  - Shows a new window with the image in it
- Every time you use imshow, the image you want to display is put in the same window so what do you do?

Read/Write Image

- Use figure command to create a new blank window
  - Then, run the imshow command to display the image on the other window
- We can also do:
  - imshow(im(:,:,1)); --→ Shows red channel
  - imshow(im(:,:,2)); --→ Shows green channel
  - imshow(im(:,:,3)); --→ Shows blue channel
- (:,:,1) means grab all of the rows and columns for the first layer (i.e. red), etc.

Read/Write Image

- When showing the red channel:
  - Darker pixels mean there isn't much red in that pixel
  - Lighter pixels mean there is a lot of red in that pixel
- Same applies for green and blue!
- How do I save images to disk? Use imwrite()
  imwrite(im, 'name_of_image.ext', 'EXT');
  - im --→ image to write to disk
  - name_of_image.ext --→ Name of the image
  - 'EXT' --→ Extension of the file (JPG, BMP, PNG, etc.)
- (:,:,1) means grab all of the rows and columns for the first layer (i.e. red), etc.

Exercise 6: Read/Write Image

- Read an Image 'Harry.jpg'
- Display the size of the image
- Displaying Colour Channels Separately
- Access 100 x 100 subimage and store it in another image
- Subtract each Image Pixel from 255 and displays it
- Writing output Images to File

Some more commands to handle images

- Convert the image into grayscale image

```
img_gray = rgb2gray(img);
```

- To find the size of an image

```
[M, N] = size(img_gray);
```

- Show Histogram

```
imhist(img_noise);
```

- Histogram Equalization

```
img_equi=histeq(img_gray,255);
```

- Show the Histogram

```
imhist(img_equi);
```

Outline

1. Automated Objects Classification System

- Automated objects classification into two classes using supervised method
- Classes
  - Circle
  - Rectangle

Supervised v/s Unsupervised

- Supervised: Task of inferring a function from labelled data. (labeled data --→ set of input features and their corresponding outputs.)
  - Classification techniques are approaches of supervised learning.
- Unsupervised: Task of inferring a function from unlabelled data.(Unlabelled data --→ set of input features withnot their corresponding outputs.)
  - Clustering techniques are the examples of unsupervised learning.

Automated Object Classification (Supervised method)

- Training Phase

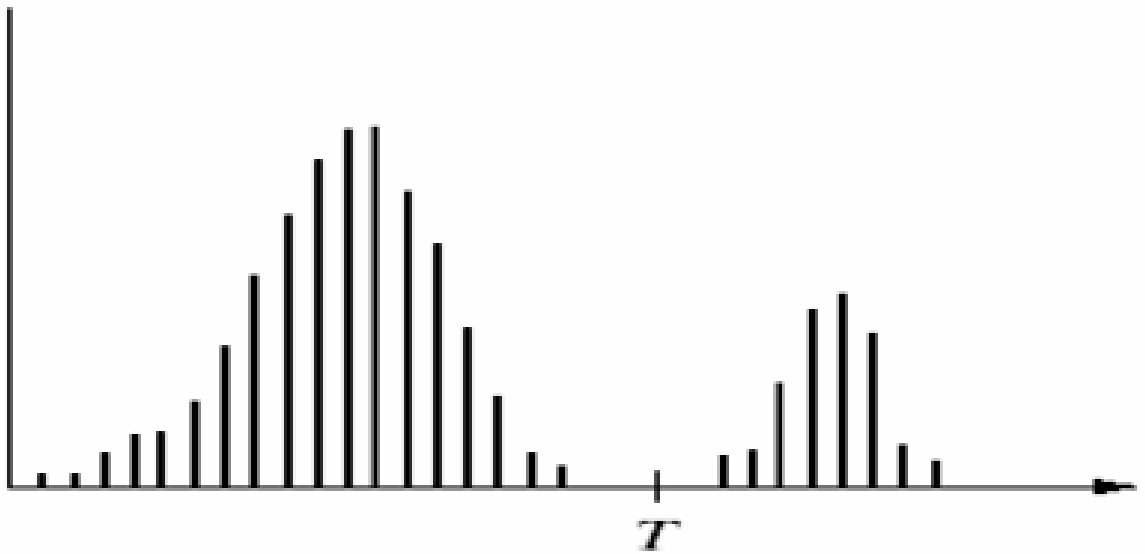
- Testing Phase

Training Phase

- Image Segmentation
  - Segmentation is to subdivide an image into its constituent regions or objects
  - Segmentation should stop when the objects of interest in an application have been isolated

Training Phase

- Image Segmentation Method (Otsu's Threshold):
  - Iterates through all the possible threshold values and calculate a measure of spread for the pixel levels at each side of the threshold.
  - Aims at finding the threshold value with minimum within class variance.



Training Phase

- Image Segmentation Method (Otsu's Threshold):
  - Read Image
    img=imread('input.bmp');
  - Show Image
    imshow(img);
  - Convert into grayscale
    img_gray=rgb2gray(img);
  - Show image
    imshow(img_gray);
  - Calculate Threshold
    T = graythresh(img_gray);
  - Convert into binary image
    img_seg = im2bw(img_gray,T);
  - Show the Segmented Image
    imshow(img_seg);

Feature extraction

- Find the Connected Components

```
cc = bwconncomp(img_seg);
```

- Calculate the Properties

```
stats = regionprops(cc, 'Area','ConvexArea', 'Eccentricity',
    'EulerNumber', 'MajorAxisLength', 'MinorAxisLength',
    'Perimeter', 'Solidity', 'Orientation');
```

- Prepare the Feature Vector

```
F=zeros(cc.NumObjects,9);
for i=1:cc.NumObjects
   F(i,1)=stats(i,1).Area;                %shape identification%
   F(i,2)=stats(i,1).ConvexArea;          %shape identification%
   F(i,3)=stats(i,1).Eccentricity;        %shape identification%
   F(i,4)=stats(i,1).EulerNumber;         %shape identification%
   F(i,5)=stats(i,1).MajorAxisLength;     %shape identification%
   F(i,6)=stats(i,1).MinorAxisLength;     %shape identification%
   F(i,7)=stats(i,1).Perimeter;           %shape identification%
   F(i,8)=stats(i,1).Solidity;            %texture identification%
   F(i,9)=stats(i,1).Orientation;         %shape identification%
end
```

Recognition using svm

- Phase 1: Training
  - Import the Training Data if Required
    load training_data;
    xdata = data(:,1:9);
    group = data(:,10:10);
  - Train the Model
    svmStruct = svmtrain(xdata,group);
- Phase 2: Testing
  - output = svmclassify(svmStruct,F);
  - Find the accuracy!!!!
    mean(double(output == ytest)) * 100

Face Recognition

- Load the image database:
  faceDatabase = imageSet('FaceDatabaseATT','recursive');
- Split Database into Training & Test Sets as (80:20)
  [*training*, *test*] = partition(faceDatabase,[0.8 0.2]);
- Extract HOG Features for single face
  person = 5;
  [*hogFeature*, *visualization*]=
  extractHOGFeatures(read(training(person),1));

Face Recognition

- Apply HOG feature extraction on training data

```
trainingFeatures =
zeros(size(training,2)*training(1).Count,4680);
featureCount = 1;
for i=1:size(training,2)
    for j = 1:training(i).Count
        trainingFeatures(featureCount,:) =
                    extractHOGFeatures(read(training(i),j));
        trainingLabelfeatureCount = training(i).Description;
        featureCount = featureCount + 1;
    end
    personIndexi = training(i).Description;
end
```

- Train the classifier using fitcecoc

```
faceClassifier = fitcecoc(trainingFeatures,trainingLabel);
```

Face Recognition

- Test the an images from Test Set

```
person = 1;
queryImage = read(test(person),2);
queryFeatures = extractHOGFeatures(queryImage);
personLabel = predict(faceClassifier,queryFeatures);
% Map back to training set to find identity
booleanIndex = strcmp(personLabel, personIndex);
integerIndex = find(booleanIndex);
```

- Display the results

```
subplot(1,2,1);
imshow(galleryImage);
title('Query Face');
subplot(1,2,2);
imshow(read(training(integerIndex),1));
title('Matched Class');
```

2. Face Detection

- MATLAB built-in class and function
  vision.CascadeObjectDetector
  Based on Viola-Jones algorithm
- Create an object for detection
  `FDetect = vision.CascadeObjectDetector;`
- Read an Image
  `I = imread('HarryPotter.jpg');`
- Apply the detection on Image
  `boundingBox = step(FDetect,I);`
- Display the detected faces

```
figure, imshow(I);
hold on
for i=1:size(boundingBox,1)
    rectangle('Position',boundingBox(i,:),'LineWidth',5,...
    'LineStyle','-','EdgeColor','r');
end
title('Face Detection');
hold off
```

3. Nose Detection

- Create an object for nose detection
  NoseDetect = vision.CascadeObjectDetector('Nose');
- Read an Image
  I = imread('HarryPotter.jpg');
- Apply the detection on Image
  boundingBox =step(NoseDetect,I);
- Display the detected noses

```
figure, imshow(I);
hold on
for i=1:size(boundingBox,1)
    rectangle('Position',boundingBox(i,:),'LineWidth',4,...
    'LineStyle','-','EdgeColor','b'); end
title('Nose Detection');
hold off
```

4. Mouth Detection

- Create an object for nose detection
  MouthDetect = vision.CascadeObjectDetector('Mouth');
- Read an Image
  I = imread('HarryPotter.jpg');
- Apply the detection on Image
  boundingBox =step(MouthDetect,I);
- Display the detected mouths

```
figure, imshow(I);
hold on
for i=1:size(boundingBox,1)
    rectangle('Position',boundingBox(i,:),'LineWidth',2,...
    'LineStyle','-','EdgeColor', 'r'); end
title('Mouth Detection');
hold off
```

5. Eye Detection

- Create an object for nose detection

```
EyeDetect = vision.CascadeObjectDetector('EyePairBig');
```

- Read an Image

```
I = imread('Harry.jpg');
```

- Apply the detection on Image

```
boundingBox =step(EyeDetect,I);
```

- Display the detected eyes

```
figure, imshow(I);
hold on
for i=1:size(boundingBox,1)
    rectangle('Position',boundingBox(i,:),'LineWidth',3,...
    'LineStyle','-','EdgeColor','b'); end
title('Eye Detection');
hold off
```

- Crop the detected eyes

```
Eyes=imcrop(I, boundingBox);
```

- Display the cropped eyes

```
figure, imshow(Eyes);
title('Cropped Eyes');
```

Outline

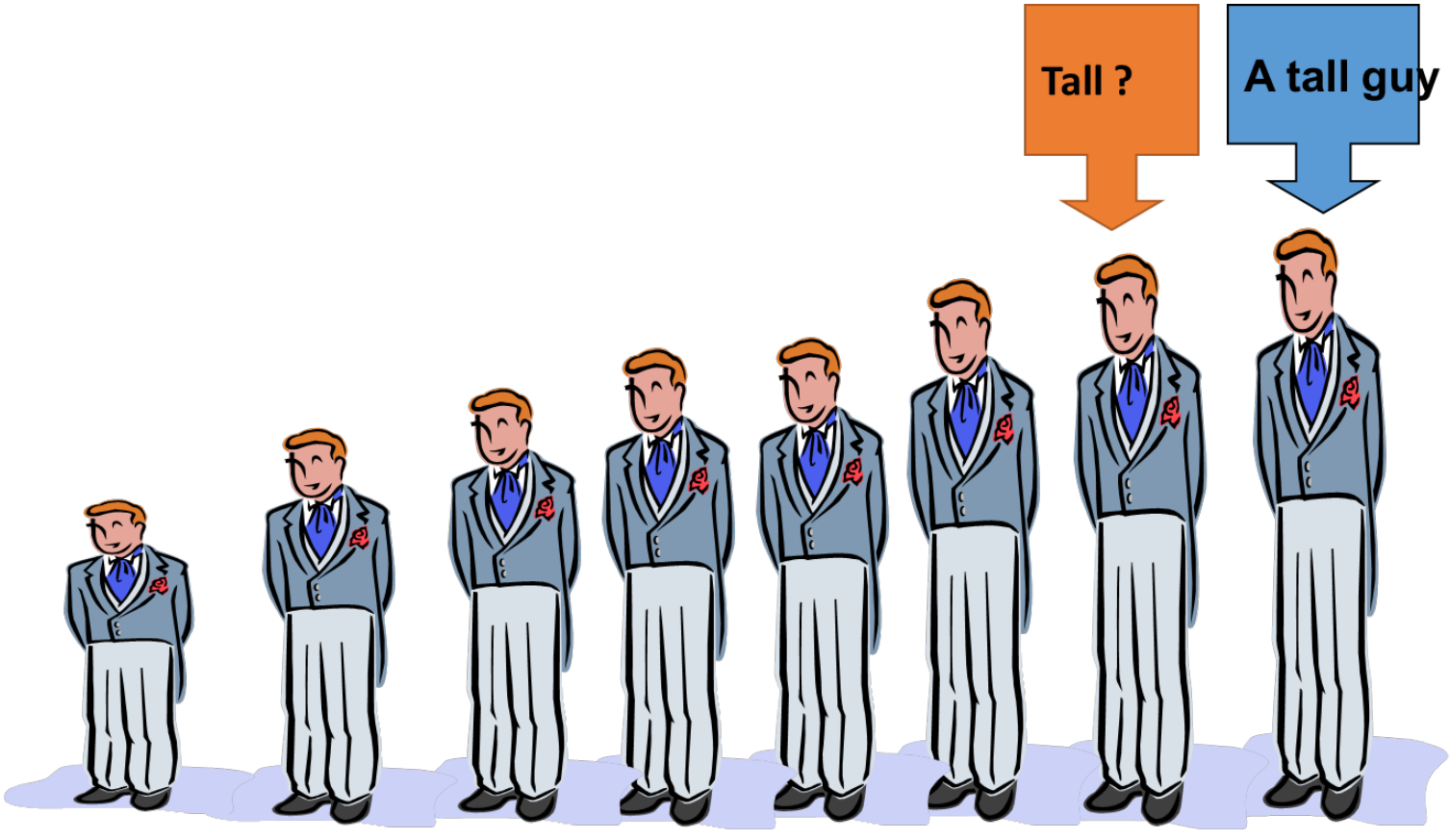
Reality is more or less uncertain, vague and ambiguous

- For example a company owner need honest person for his company.
- Available choices can be: very honest, honest, dishonest
- In computer: two choices are possible, honest and dishonest.

Color of Rose

WHAT ABOUT THIS???

Handsome



Where do tall people start?

Natural phrases have imprecise meaning

- Dinosaurs ruled the earth for a long period.
- It has not rained for long period.
- I had to wait for bus for a long period.
- Today I woke up early.

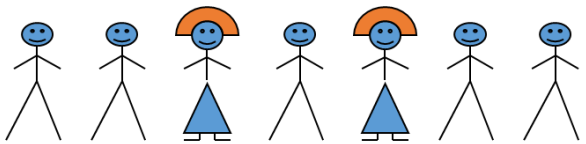
Humans are good in handling imprecision but computers are not

- If obstacle is close then apply break
- If it is hot outside then dont go outside.

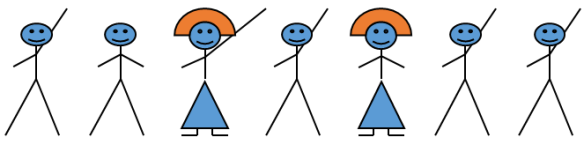
Purpose of Fuzzy Set

To generalize the classical set to a set (fuzzy set) which can capture imprecise notion like good, bad, hot , beautiful etc.

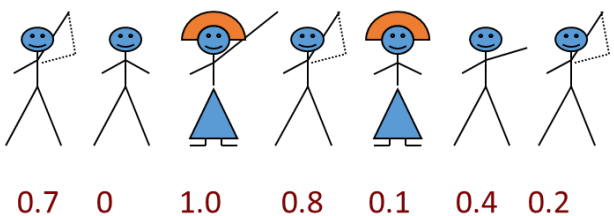
An Example of Fuzzy Set



- A class of students
  (e.g. B.Tech. Students taking "Fuzzy Logic subject")
- The universe of discourse: X



- "Who does have a driver's licence?"
- A subset of X = A (Crisp) Set
- $\chi(X)$ = CHARACTERISTIC FUNCTION



- "Who can drive very well?"
  $\mu(X)$ = MEMBERSHIP FUNCTION

CLASSICAL / CRISP SETS

- A well defined collection of objects is called a set.
  - Examples:
    - Set of Integers
    - Set of Real Numbers
    - Set of Rational Numbers

Characteristic Function for Crisp Set

When a set 'A' is a crisp set then, the membership/degree of belongingness of each element to the set 'A' can take only two values 0 and 1 i.e.,

$\chi_A(x) = 1$ if and only if x belongs to A

$\chi_A(x) = 0$ if and only if x does not belongs to A

i.e .χ: X →{0, }1

Membership Function of A Fuzzy Set

- In fuzzy sets ,each element is mapped to [0,1] by a membership function

$$\mu_A : x \rightarrow [0,1],$$

including 0 and 1.

- For universal set X, a fuzzy set is defined as

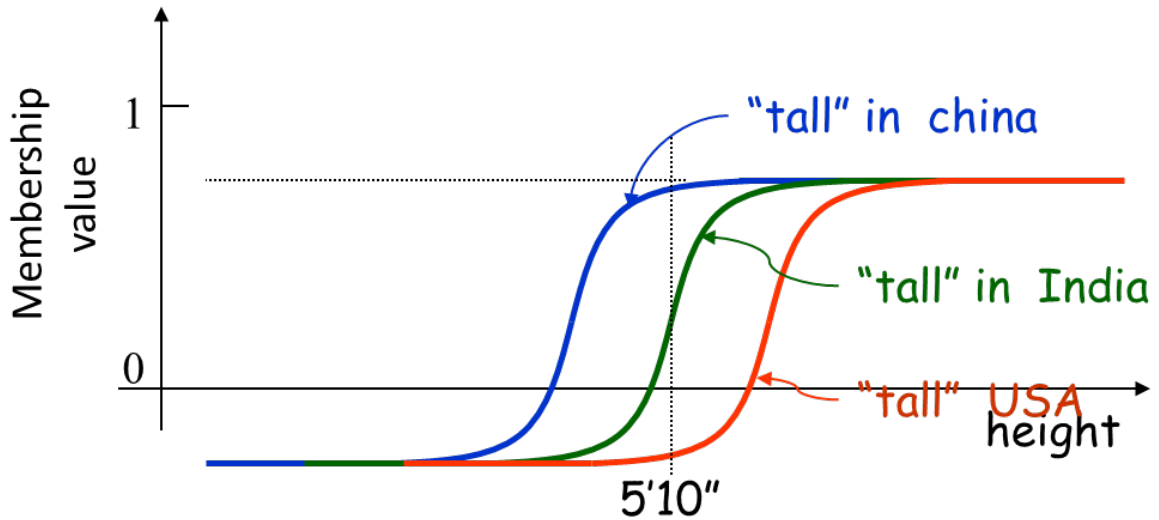
$A = \{(x, \mu_A(x)) : x \in X\}$

e.g.

C={(Delhi,0.9);(Mumbai,0.8);(Chennai,0.6)}

Membership Functions (MFs)

- A fuzzy set is completely characterized by a membership function.
  - a subjective measure.
  - *not* a probability measure.



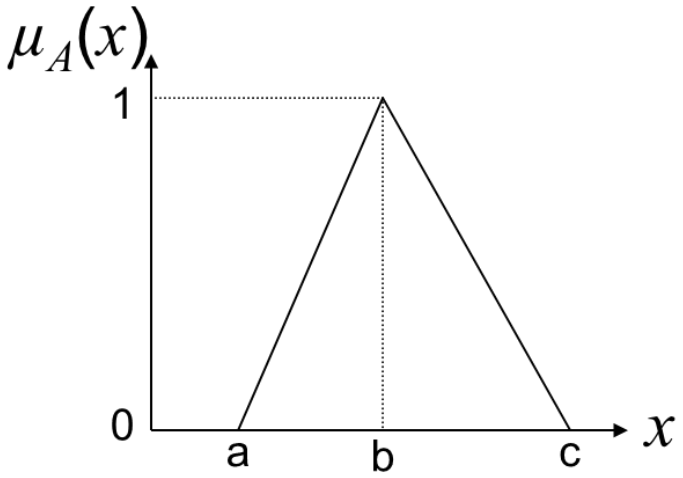
Fuzzy Versus Probability

- A bottle of liquid has a probability of 0.9 of being rat poison.
- A second bottle contains *lots* of rat poison i.e., 0.9.
- The meaning of 0.9 for the two bottles clearly differs significantly and would impact your choice should you be dying of thirst.

Triangular Membership Function

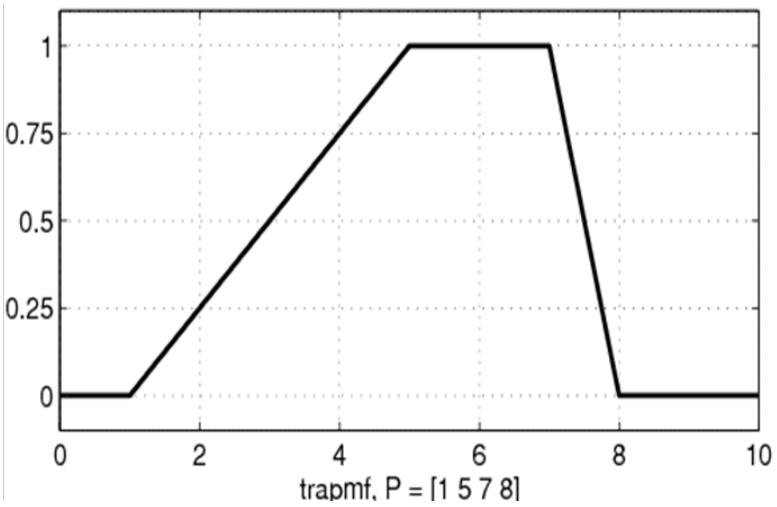
- a, b and c represent the *x* coordinates.
- (a: lower boundary and c: upper boundary where membership degree is zero, b: the centre where membership degree is 1)

$$\mu_A(x) = \begin{cases} 0 & if\ x \le a \\ \dfrac{x-a}{b-a} & if\ a \le x \le b \\ \dfrac{c-x}{c-b} & if\ b \le x \le c \\ 0 & if\ x \ge c \end{cases}$$



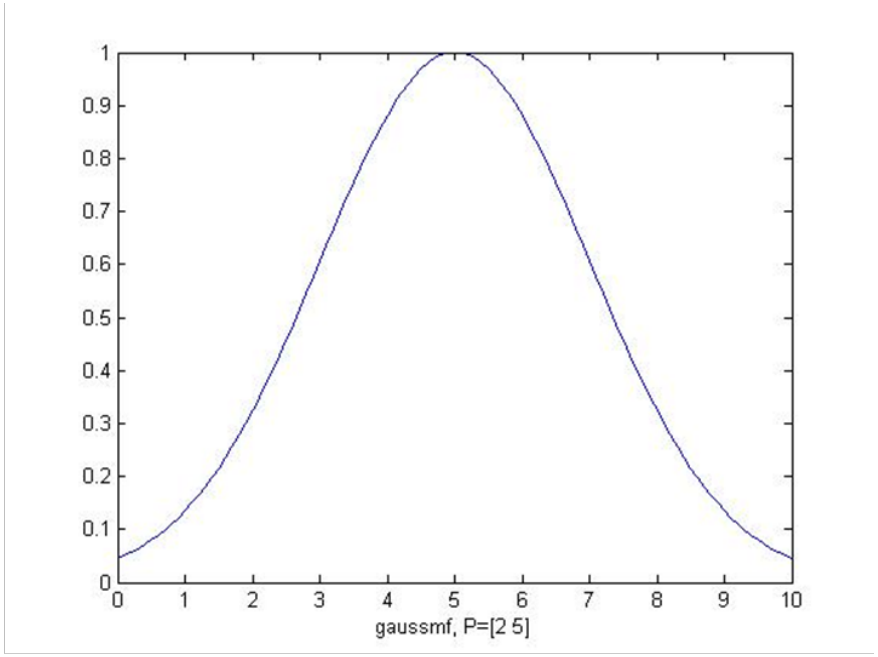
Trapezoidal Membership Function

$$f(x;a,b,c,d) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ 1, & b \le x \le c \\ \dfrac{d-x}{d-c}, & c \le x \le d \\ 0, & d \le x \end{cases}$$



Gaussian Membership Function

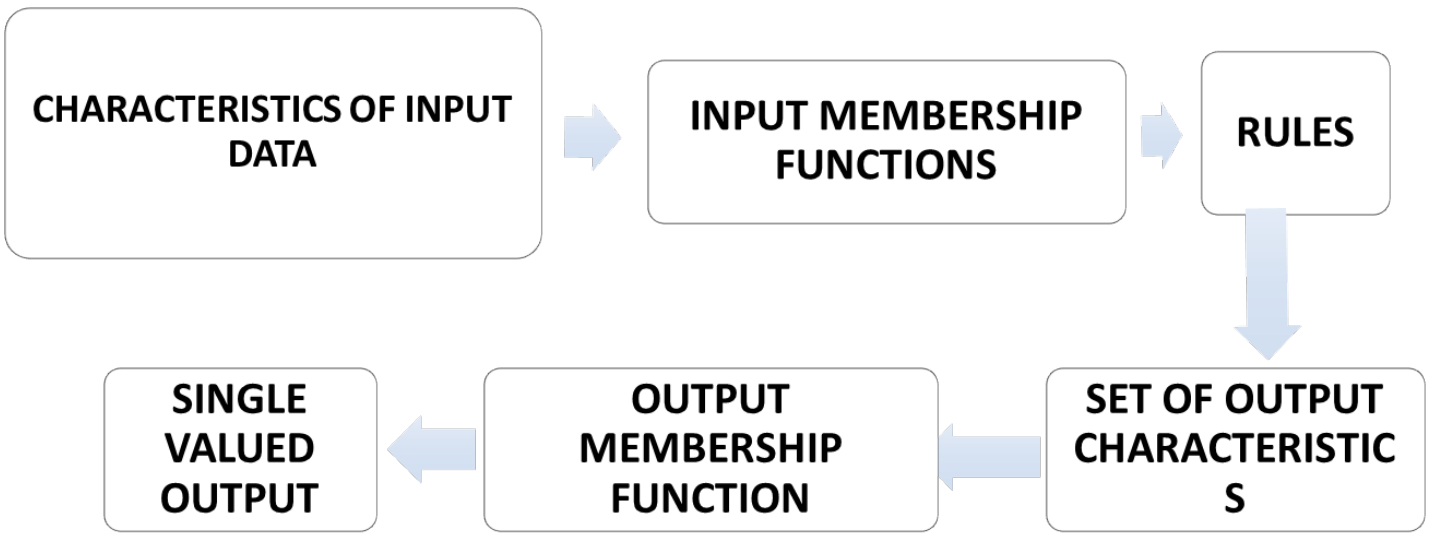
$$f_{gmf}\left(x;\sigma,c\right)=e^{\frac{-(x-c)^2}{2\sigma^2}}$$



C=5= centre
Sigma= 2=width of the function

Fuzzy Inferece System

The process of formulating the mapping from a given input to an output using fuzzy logic is called the fuzzy inference.



Outline

TIPPING PROBLEM

- Goal: To decide appropriate **tip** on the basis of **service quality** and **food quality** in a restaurant.
- Service and food quality are given number between 0(poor) to 10 (excellent).
- Assume that tip ranges from 5% (cheap) to 25% (generous)

STEP 1:Fuzzification of input and output

- Service [0-10] is divided into three fuzzy sets poor, good and excellent.
- Food quality [0-10] is divided into two fuzzy sets rancid and delicious.
- Tip 5% to 25%, is divided into three fuzzy sets cheap, generous and average

STEP 2 FUZZY RULE BASE

Based on experience of restaurants following are rules of tipping *:

1. *If the service is poor or the food is rancid, then tip is cheap.*
2. *If the service is good, then tip is average.*
3. *If the service is excellent or the food is delicious, then tip is generous.*

**Subject to local tradition and cultural bias.*

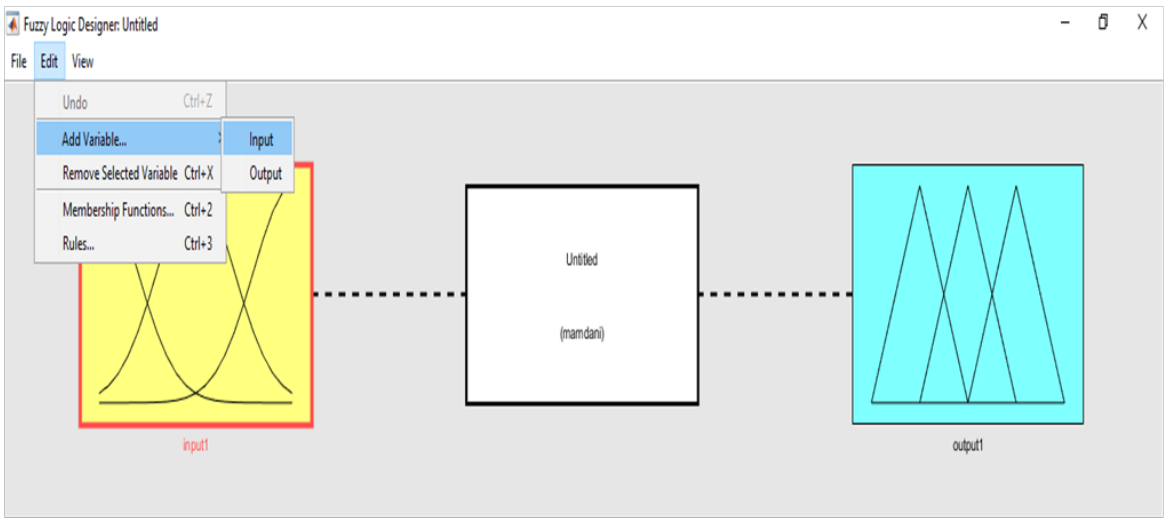
STEP 3 MATLAB IMPLEMENTATION-1

Type following in MATLAB command window/prompt

>> fuzzy

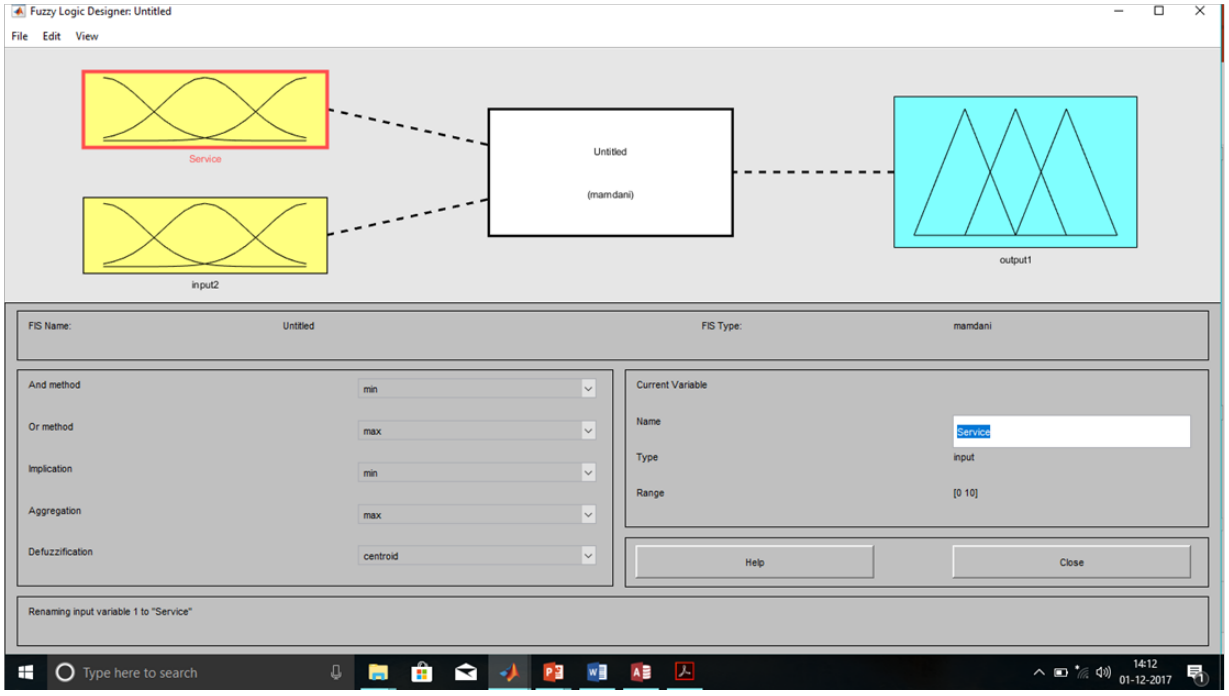
- Select **Edit** > **Add variable** > **Input**.

  A second yellow box labeled **input2** appears.



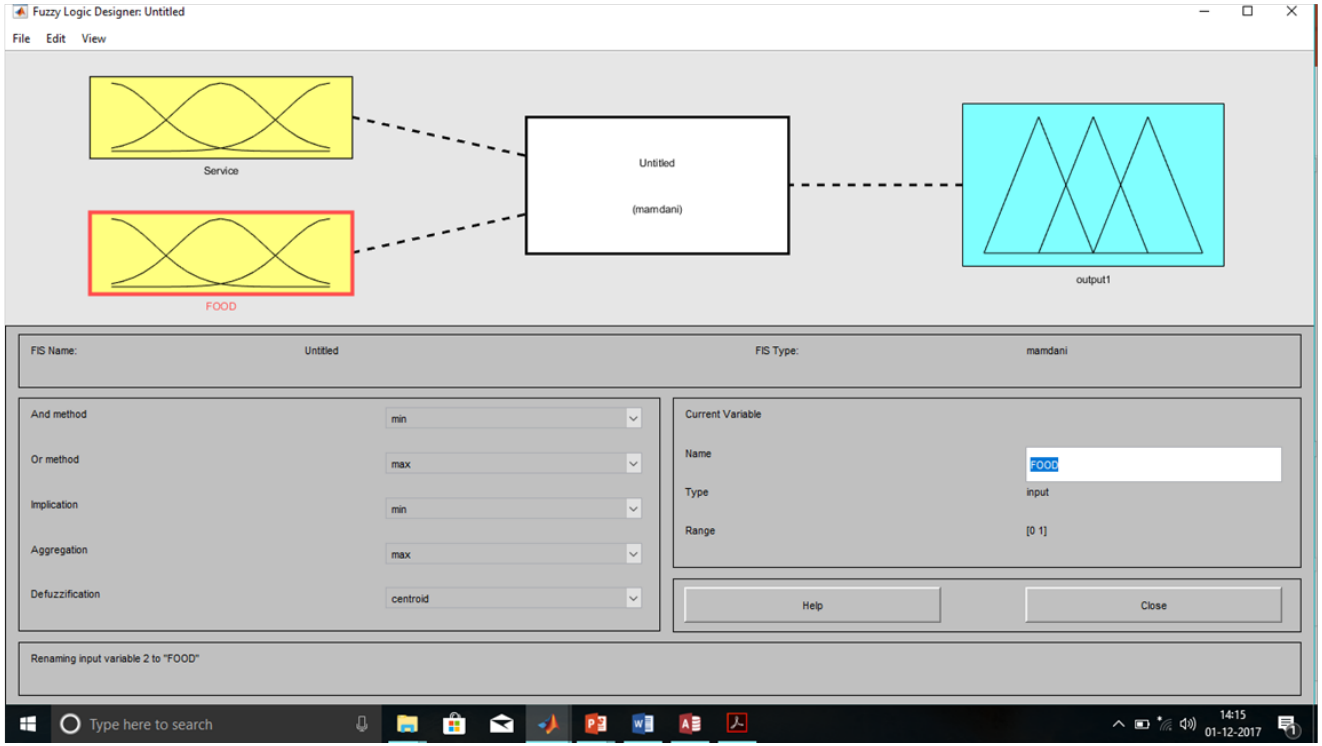
STEP 3 MATLAB IMPLEMENTATION-2

- A second yellow box labeled **input2** appears.
- Click the yellow box **input1**. This box is highlighted with a red outline.
- Edit the **Name** field from input1 to service, and press **Enter**.



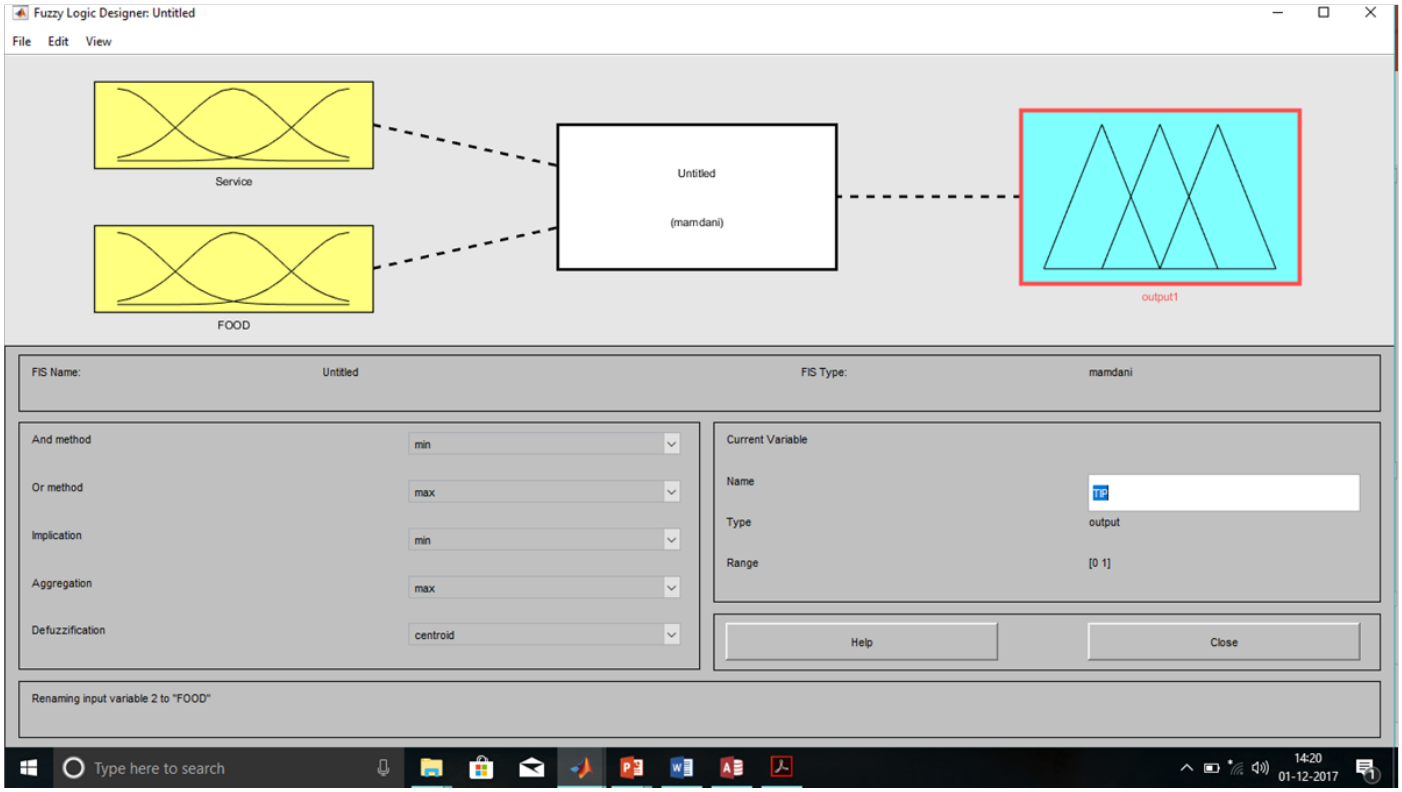
STEP 3 MATLAB IMPLEMENTATION-3

- Click the yellow box **input2**. This box is highlighted with a red outline.
- Edit the **Name** field from input2 to food, and press **Enter**.



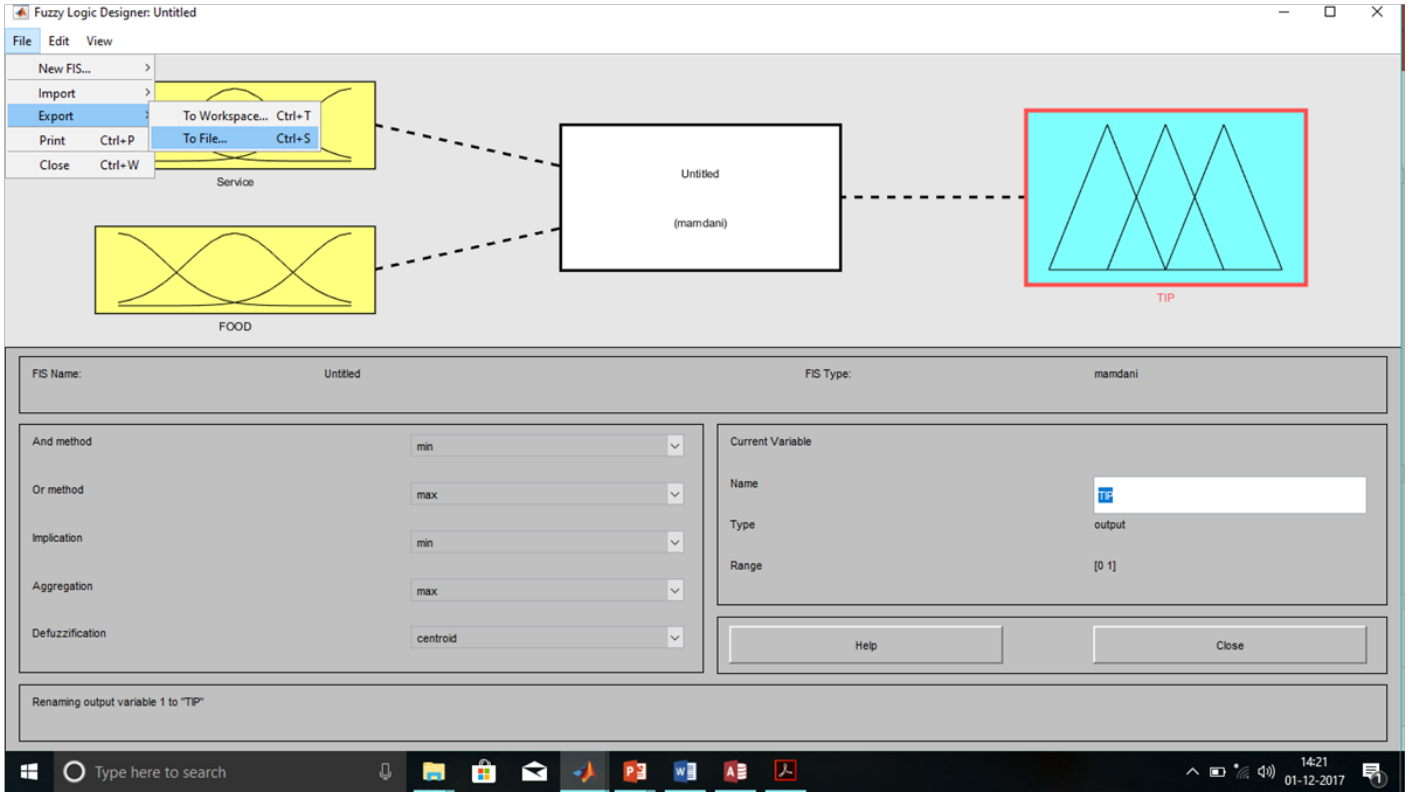
STEP 3 MATLAB IMPLEMENTATION-4

- Click the blue box **output1**.
- Edit the **Name** field from output1 to tip, and press **Enter**.



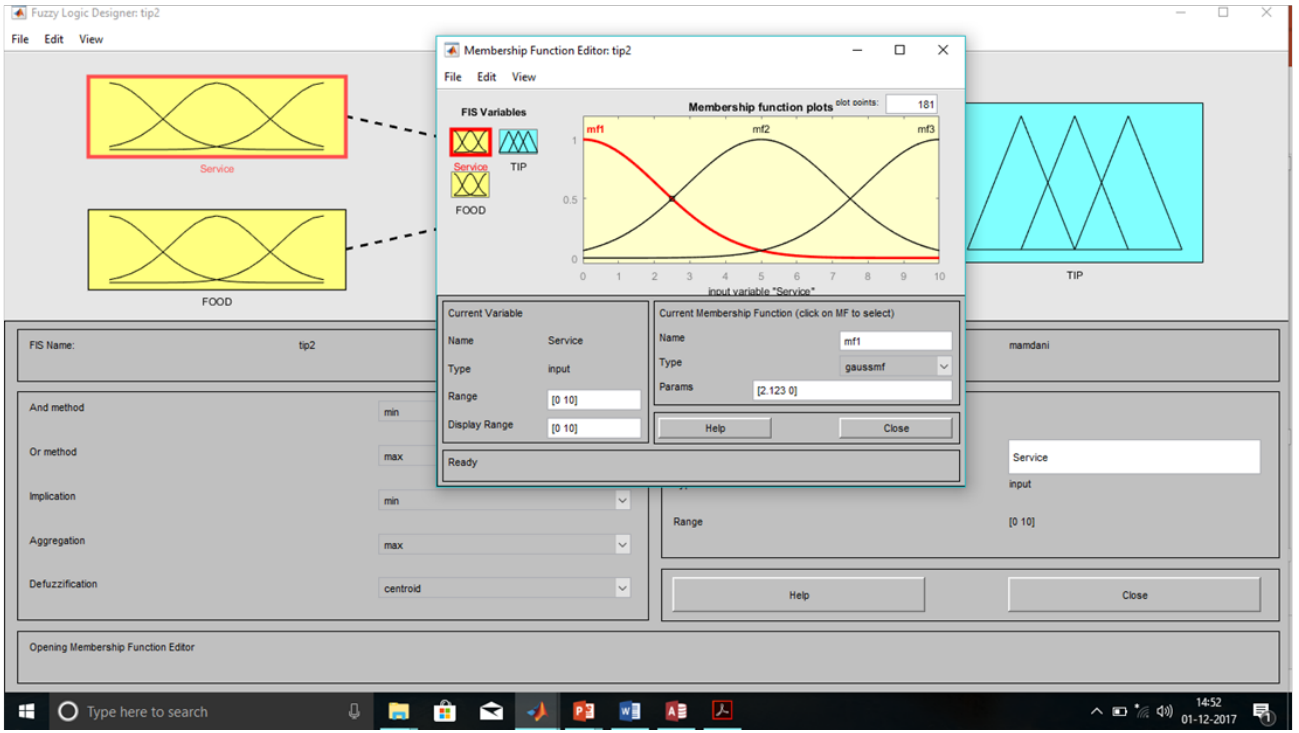
STEP 3 MATLAB IMPLEMENTATION-5

- Select **File** > **Export** > **To File**
- Save to desktop with name tip.



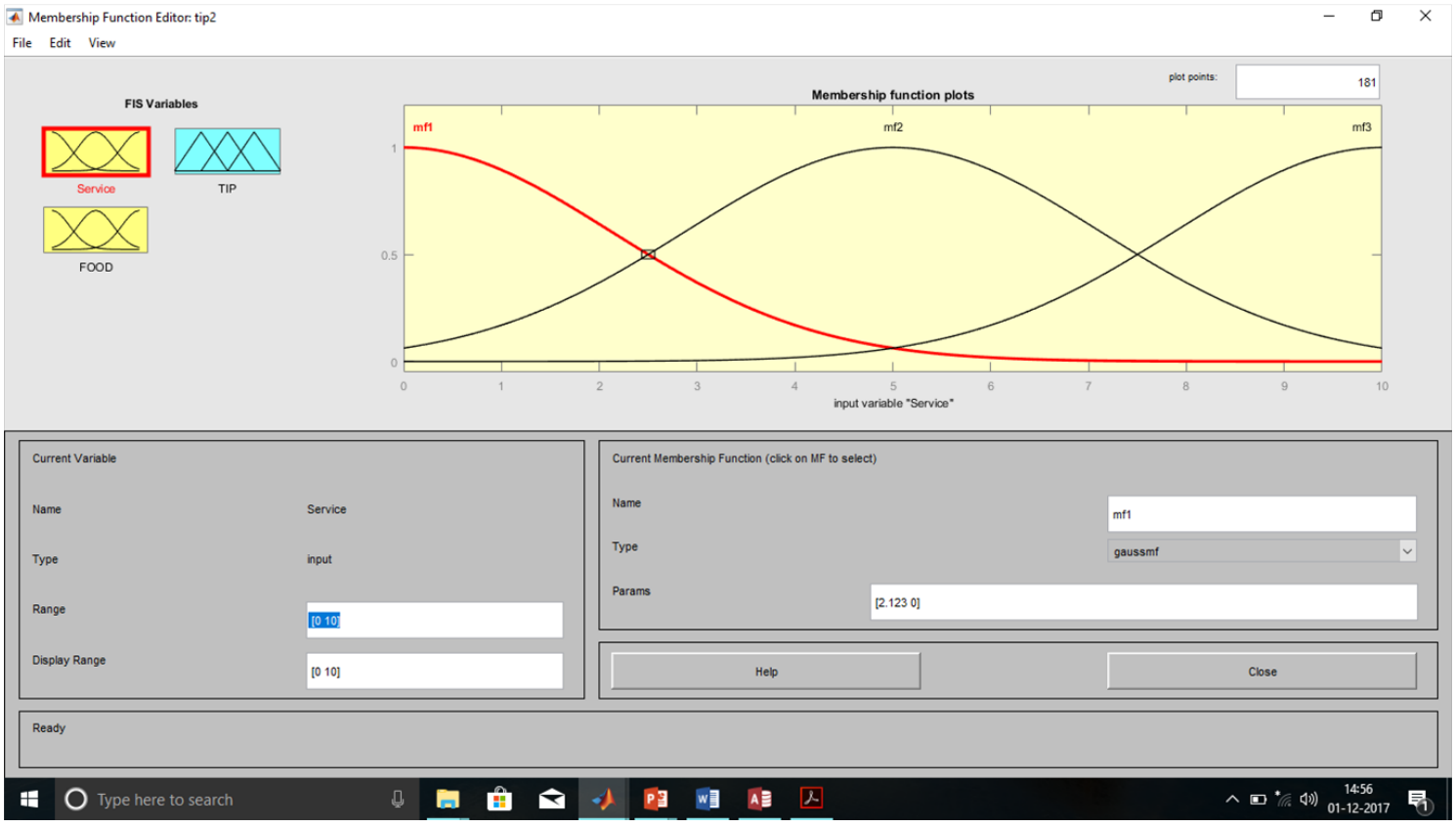
STEP 3 MATLAB IMPLEMENTATION-6

- In **Fuzzy Logic Designer** window.
- Double-click the input variable service to open the Membership Function Editor.



STEP 3 MATLAB IMPLEMENTATION-7

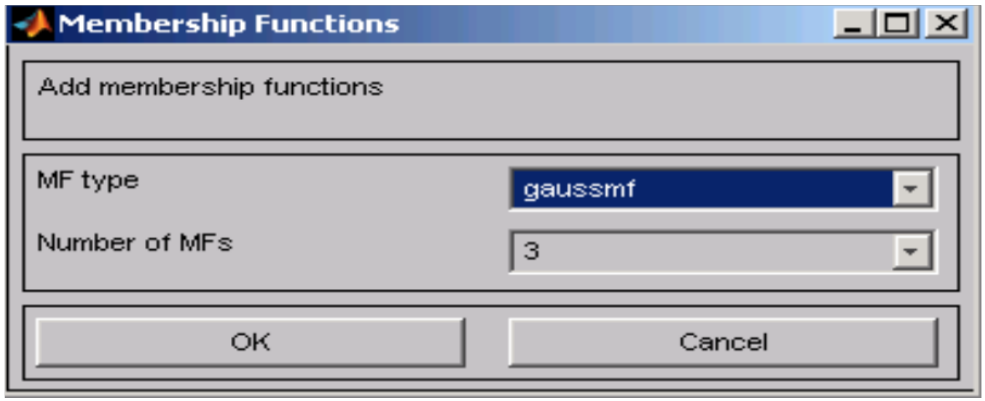
- In the Membership Function Editor, enter [0 10] in the **Range** and the **DisplayRange** fields.



STEP 3 MATLAB IMPLEMENTATION-8

❖ Create membership functions for the input variable service.

- Select **Edit** > **Remove All MFs** to remove the default membership functions for the input variable service.
- Select **Edit** > **Add MFs** to open the Membership Functions dialog box.
- In the Membership Functions dialog box, select gaussmf as the **MF Type**.
- Verify that 3 is selected as the **Number of MFs**.
- Click **OK** to add three Gaussian curves to the input variable service.



STEP 3 MATLAB IMPLEMENTATION-9

❖ Rename the membership functions for the input variable service, and specify their parameters.

➢Click on the curve named mf1 to select it, and specify the following fields in the **Current Membership Function (click on MF to select)** area:

- In the **Name** field, enter poor.
- In the **Params** field, enter [1.5 0].

➢Click on the curve named mf2 to select it, and specify the following fields in the **Current Membership Function (click on MF to select)** area:

- In the **Name** field, enter good.
- In the **Params** field, enter [1.5 5].

➢ Click on the curve named mf3, and specify the following fields in the **Current Membership Function (click on MF to select)** area:

- In the **Name** field, enter excellent.
- In the **Params** field, enter [1.5 10].

STEP 3 MATLAB IMPLEMENTATION 10

- Create the membership functions for the input variable food.
- Select **Edit** > **Remove All MFs** to remove the default Membership Functions for the input variable food.
- Select **Edit** > **Add MFs** to open the Membership Functions dialog box.
- In the Membership Functions dialog box, select trapmf as the **MF Type**.
- Select 2 in the **Number of MFs** drop-down list.
- Click **OK** to add two trapezoidal curves to the input variable food.

STEP 3 MATLAB IMPLEMENTATION-11

❖Rename the membership functions for the input variable food, and specify their parameters:

- In the **FIS Variables** area, click the input variable food to select it.
- Click on the curve named mf1, and specify the following fields in the **Current Membership Function (click on MF to select)** area:
  - In the **Name** field, enter rancid.
  - In the **Params** field, enter [0 0 1 3].
- Click on the curve named mf2 to select it, and enter delicious in the **Name** field and set [0 2 3 3] in the **Params** field.

STEP 3 MATLAB IMPLEMENTATION-12

❖ Click on the output variable tip to select it.

- Enter [0 30] in the **Range** and the **Display Range** fields to cover the output range. (The inputs ranges from 0 to 10, but the output is a tip between 5% and 25%.)
- Rename the default triangular membership functions for the output variable tip, and specify their parameters.

STEP 3 MATLAB IMPLEMENTATION-13

- Click the curve named mf1 to select it, and specify the following fields in the **Current Membership Function (click on MF to select)** area:
  - In the **Name** field, enter cheap.
  - In the **Params** field, enter [0 5 10].
- Click the curve named mf2 to select it, and specify the following fields in the **Current Membership Function (click on MF to select)** area:
  - In the **Name** field, enter average.
  - In the **Params** field, enter [10 15 20].
- Click the curve named mf3 to select it, and specify the following:
  - In the **Name** field, enter generous.
  - In the **Params** field, enter [20 25 30].

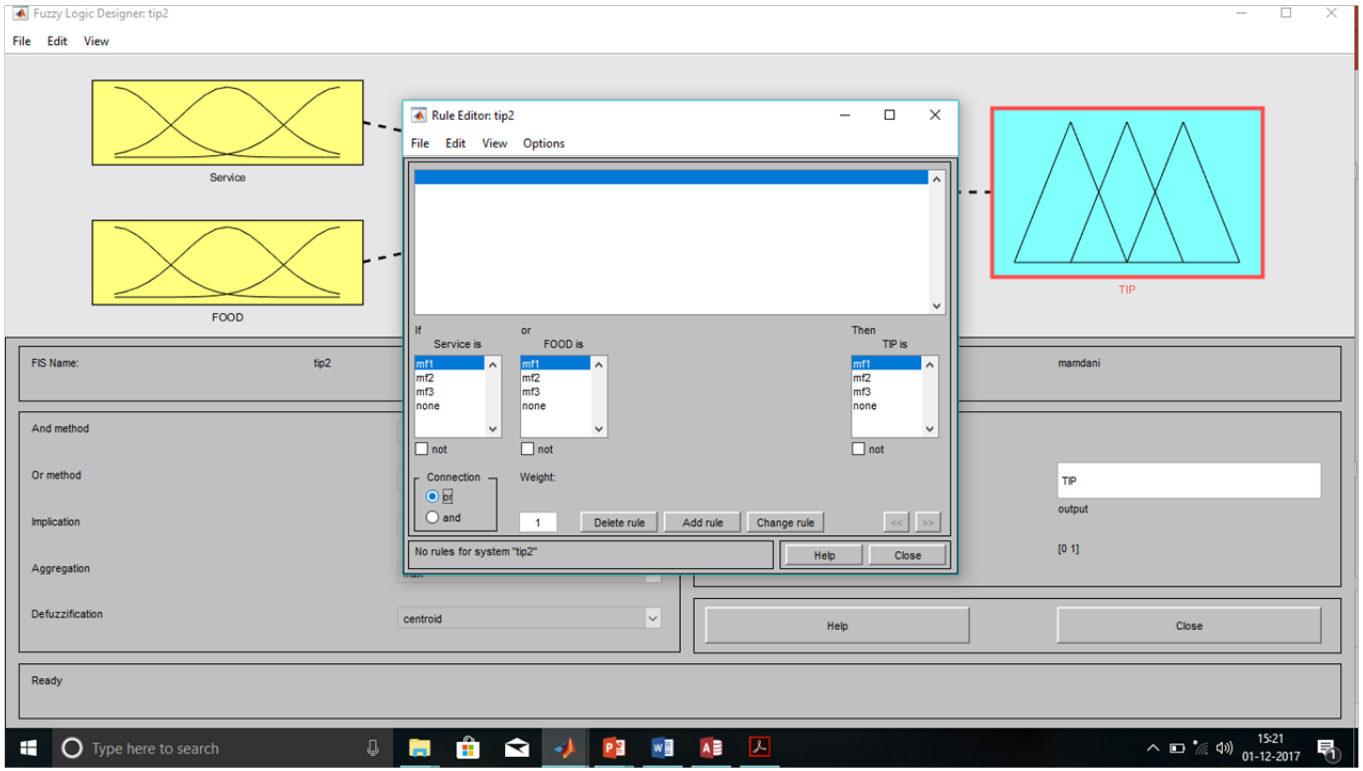
STEP 3 MATLAB IMPLEMENTATION-13

- Click the curve named mf1 to select it, and specify the following fields in the **Current Membership Function (click on MF to select)** area:
  - In the **Name** field, enter cheap.
  - In the **Params** field, enter [0 5 10].
- Click the curve named mf2 to select it, and specify the following fields in the **Current Membership Function (click on MF to select)** area:
  - In the **Name** field, enter average.
  - In the **Params** field, enter [10 15 20].
- Click the curve named mf3 to select it, and specify the following:
  - In the **Name** field, enter generous.
  - In the **Params** field, enter [20 25 30].

STEP 3 MATLAB IMPLEMENTATION-14

- Rule Editor

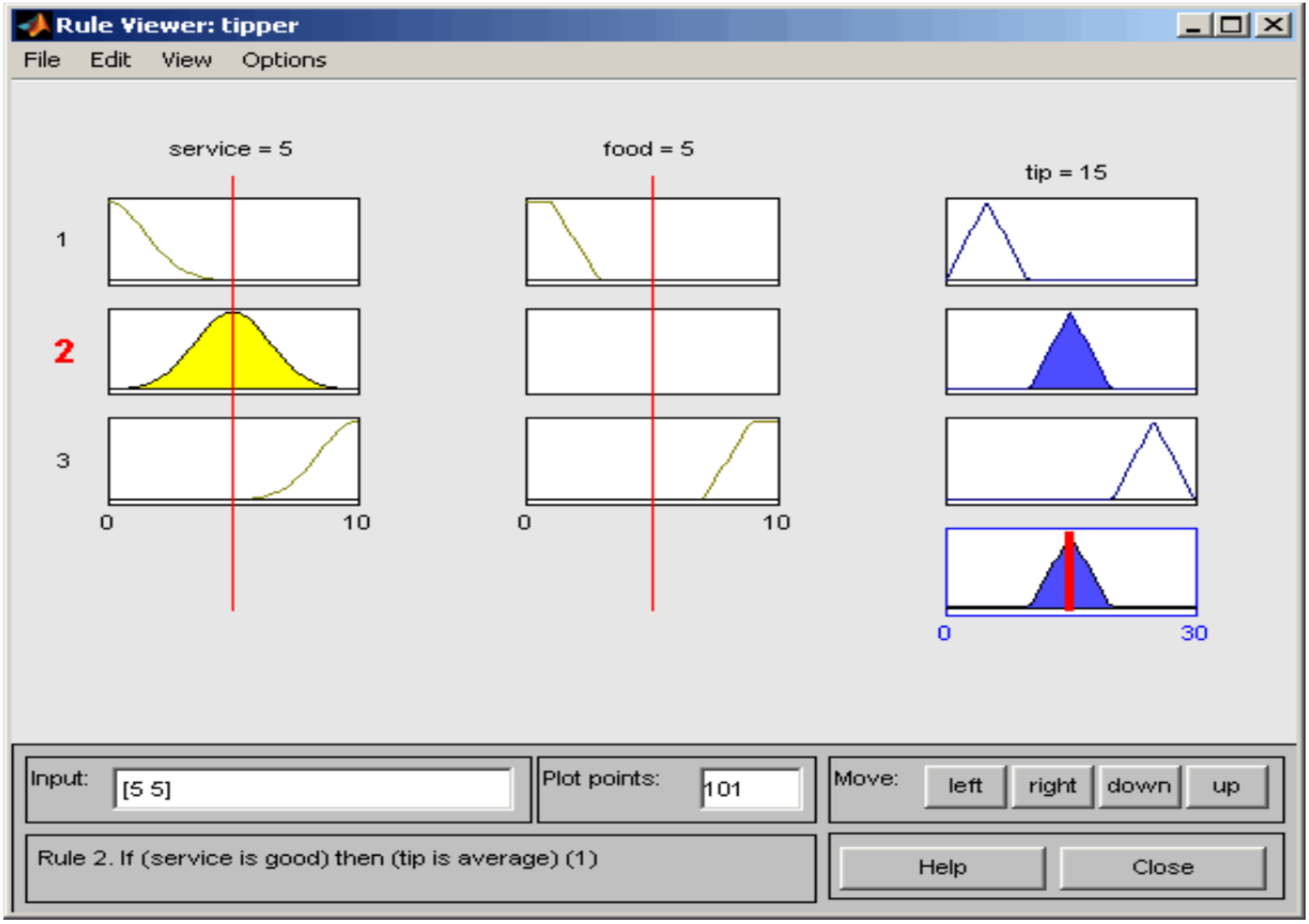
  go to the **Edit** menu and select **Rules**



Rules

1. *If (service is poor) or (food is rancid) then (tip is cheap) (1)*
2. *.If (service is good) then (tip is average) (1)*
3. *If (service is excellent) or (food is delicious) then (tip is generous) (1)*

Rule Viewer


Rule Viewer: tipper
File Edit View Options
service = 5
food = 5
tip = 15
1
2
3
0
10
0
10
0
30
Input:
[5 5]
Plot points:
101
Move:
left
right
down
up
Rule 2. If (service is good) then (tip is average) (1)
Help
Close

Thank You