**PATTERN RECOGNITION & MACHINE LEARNING (PRML) PROJECTS ON VARIOUS DATASETS USING MATLAB SOFTWARE**

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **Lab No.** | **Title** | **Software** | **Page No.** |
| 1. | Printed Alphabet Recognition using Image Dataset | MATLAB | 3-9 |
| 2. | Naïve Baye’s Classification on Iris and Alphabet Dataset | MATLAB | 10-12 |
| 3. | Implementing K-means & Hierarchical Clustering | MATLAB | 13-17 |
| 4. | Implementing AND gate using Perceptron | MATLAB | 18-21 |
| 5. | Implementing EX-OR Gate using Backpropagation | MATLAB | 22-23 |
| 6. | Classification of A & B English letters using Backpropagation | MATLAB | 24-27 |
| 7. | SVM for EX-OR | MATLAB | 28-21 |
| 8. | SVM for English Alphabets | MATLAB | 32-34 |
| 9. | PCA for inbuilt data set ‘ingredients’ | MATLAB | 35 |
| 10. | PCA for Alphabet Dataset | MATLAB | 36-37 |
| 11. | Understanding CNN Algorithm | -NA- | 38-41 |

**EXPERIMENT NO.1**

**Aim-** To perform Printed Alphabet Recognition on image dataset

**Title**- Printed Alphabet Recognition on MATLAB

**Task**- Classification of all 5 alphabet (A, B, C, D, E)

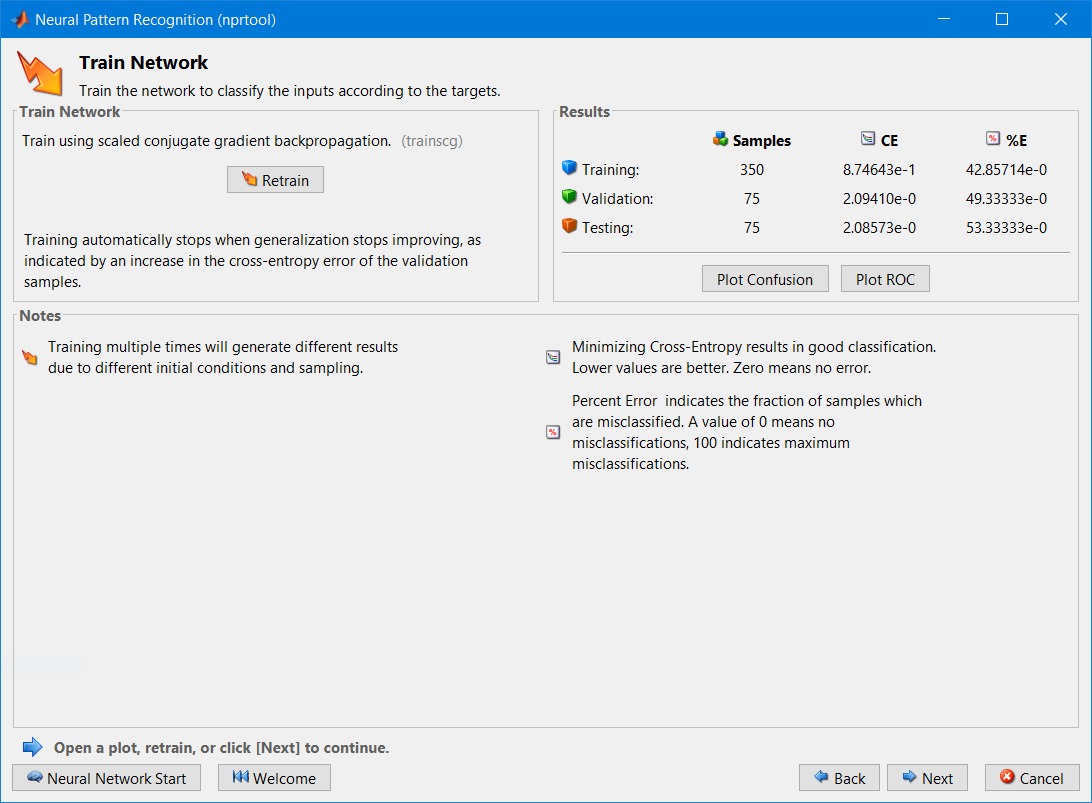
**Dataset used -**

Given below are 2 images for each alphabet from our image dataset.

C:\Users\User\Desktop\Alphabet Database\A\12.png C:\Users\User\Desktop\Alphabet Database\A\10.png C:\Users\User\Desktop\Alphabet Database\B\10.png C:\Users\User\Desktop\Alphabet Database\B\3.png C:\Users\User\Desktop\Alphabet Database\C\5.png C:\Users\User\Desktop\Alphabet Database\C\5.png C:\Users\User\Desktop\Alphabet Database\D\6.png C:\Users\User\Desktop\Alphabet Database\D\14.png C:\Users\User\Desktop\Alphabet Database\E\11.png C:\Users\User\Desktop\Alphabet Database\E\4.png

I have prepared a dataset with 100 image of each alphabet, so a total of 500 image samples for our dataset.

Out of this we have used 15 % for test set, 15% for validation set and 70% for training data. As you can see in our nprtool we have 350 images for Training set, 75 images for validation set and 75 images for test set.



**Neural network configuration** -

We have 24 input nodes for 6 x 4 =24 pixels = 24 features

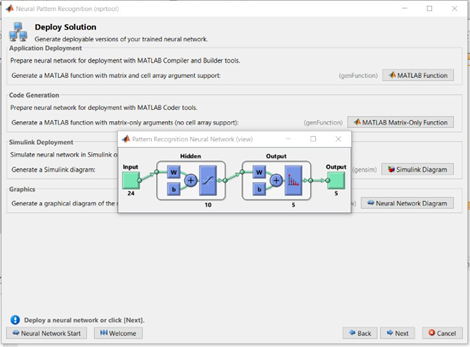
We have 5 output nodes as we have 5 different values representing 5 different alphabets

We have 10 nodes in each of the hidden layers (2)

Each layer (input, hidden, output) is denoted by weights.

As this is an example of supervised learning, we know the correct output is known, and the output is result our Deep learning NN classification.

The ‘correct output’ – ‘output’ = error, this error is passed back to the hidden layer to adjust weights of the nodes



**Code Explanation-**

Firstly, all the file locations were put in the code. Then a function cellmat() was used which takes 4 integer values and 1 scaler value. Here the scalar value is 1 and integer values have been included/given. Then for each alphabets the size of the pictures was resized to (32,42). This was done so as to get 24 features for each alphabets. The rest of the code is for the grey scale image conversion. In the end the mean of all the results in accordance to the training data was taken. In addition to that a validation set was created which was a completely unseen data and the model was implemented on that data.

The accuracy chart for training, validation and test is given below:

Below is the curve of the AUC-ROC which is a performance measurement for classification problem at various threshold. It tells us how much he model is capable of distinguishing the classes. It is a graph between False Positive rate (the predicted value is true/positive but the actual value is negative) and True Positive Rate (The predicted and the actual value is true).

**Code to run on MATLAB Software (Classification using Neural Network Back Propagation) :**

clc;

clear all;

close all;

A\_folder = dir('E:\Notes\SEM 7\PRML\Alphabet Database\A\\*.png');

B\_folder = dir('E:\Notes\SEM 7\PRML\Alphabet Database\B\\*.png');

C\_folder = dir('E:\Notes\SEM 7\PRML\Alphabet Database\C\\*.png');

D\_folder = dir('E:\Notes\SEM 7\PRML\Alphabet Database\D\\*.png');

E\_folder = dir('E:\Notes\SEM 7\PRML\Alphabet Database\E\\*.png');

A\_result=cellmat(1,length(A\_folder),32,42);

B\_result=cellmat(1,length(B\_folder),32,42);

C\_result=cellmat(1,length(C\_folder),32,42);

D\_result=cellmat(1,length(D\_folder),32,42);

E\_result=cellmat(1,length(E\_folder),32,42);

m=100;

for i=1:m

A\_result{1,i}=imresize(imread(strcat('E:\Notes\SEM 7\PRML\Alphabet Database\A\',A\_folder(i).name)),[32 42]);

A\_DB=A\_result{1,i};

for j=1:32

for k=1:42

if A\_DB(j,k)>=157

A\_DB(j,k)=1;

else

A\_DB(j,k)=0;

end

end

end

A\_result{1,i}=A\_DB;

A\_result{1,i}=reshape(A\_result{1,i},[],24);

B\_result{1,i}=imresize(imread(strcat('E:\Notes\SEM 7\PRML\Alphabet Database\B\',B\_folder(i).name)),[32 42]);

B\_DB=B\_result{1,i};

for j=1:32

for k=1:42

if B\_DB(j,k)>=157

B\_DB(j,k)=1;

else

B\_DB(j,k)=0;

end

end

end

B\_result{1,i}=B\_DB;

B\_result{1,i}=reshape(B\_result{1,i},[],24);

C\_result{1,i}=imresize(imread(strcat('E:\Notes\SEM 7\PRML\Alphabet Database\C\',C\_folder(i).name)),[32 42]);

C\_DB=C\_result{1,i};

for j=1:32

for k=1:42

if C\_DB(j,k)>=157

C\_DB(j,k)=1;

else

C\_DB(j,k)=0;

end

end

end

C\_result{1,i}=C\_DB;

C\_result{1,i}=reshape(C\_result{1,i},[],24);

D\_result{1,i}=imresize(imread(strcat('E:\Notes\SEM 7\PRML\Alphabet Database\D\',D\_folder(i).name)),[32 42]);

D\_DB=D\_result{1,i};

for j=1:32

for k=1:42

if D\_DB(j,k)>=157

D\_DB(j,k)=1;

else

D\_DB(j,k)=0;

end

end

end

D\_result{1,i}=D\_DB;

D\_result{1,i}=reshape(D\_result{1,i},[],24);

E\_result{1,i}=imresize(imread(strcat('E:\Notes\SEM 7\PRML\Alphabet Database\E\',E\_folder(i).name)),[32 42]);

E\_DB=E\_result{1,i};

for j=1:32

for k=1:42

if E\_DB(j,k)>=157

E\_DB(j,k)=1;

else

E\_DB(j,k)=0;

end

end

end

E\_result{1,i}=E\_DB;

E\_result{1,i}=reshape(E\_result{1,i},[],24);

A(i,:)=mean(A\_result{1,i});

B(i,:)=mean(B\_result{1,i});

C(i,:)=mean(C\_result{1,i});

D(i,:)=mean(D\_result{1,i});

E(i,:)=mean(E\_result{1,i});

end

X{1,1}=A;

X{1,2}=B;

X{1,3}=C;

X{1,4}=D;

X{1,5}=E;

X=cell2mat(reshape(X,[5 1]));

Y=zeros(500,5);

Y(1:100,1)=1;

Y(101:200,2)=1;

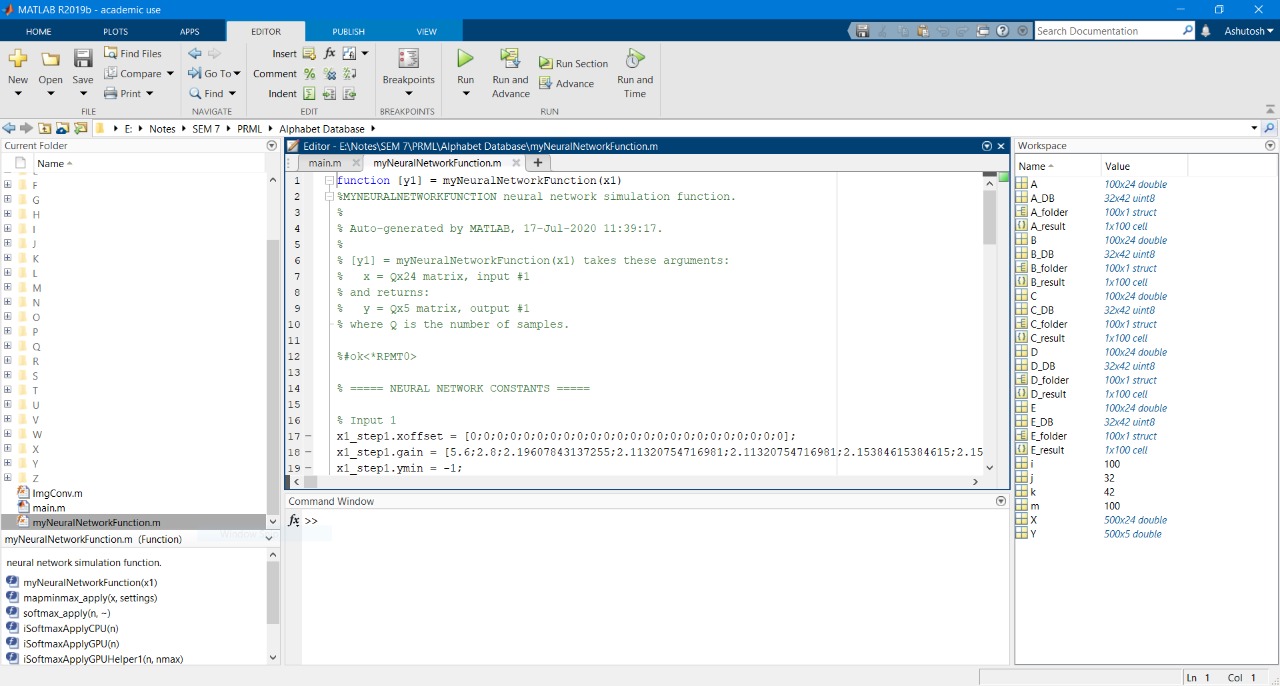
Y(201:300,3)=1;

Y(301:400,4)=1;

Y(401:500,5)=1;

**Next step after execution of the code** - Run 'nnstart' in command window

We observe this **MATLAB generated function** with back-propagation, which is the essence of neural net training. It is the method of fine-tuning the weights of a neural net based on the error rate obtained in the previous epoch. is an important mathematical tool for improving the accuracy.



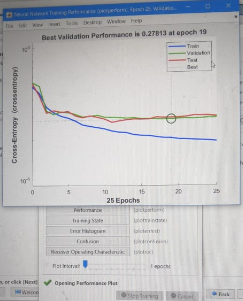
**Output observed** –

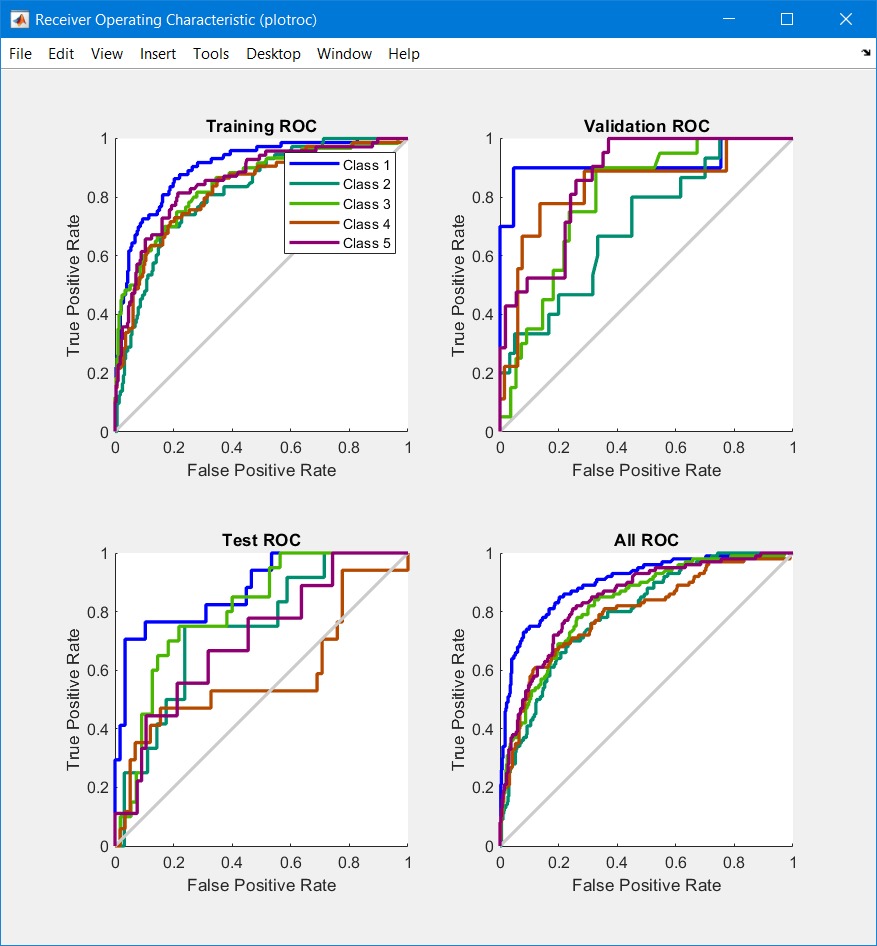
We obtained the graph plotted between Cross- Entropy and the 25 Epochs.

Epochs basically indicates the number of passes of the entire training dataset the machine learning algorithm has completed. An epoch refers to one cycle through the full training dataset, we used 25 epochs as too many of them can lead to overfitting.

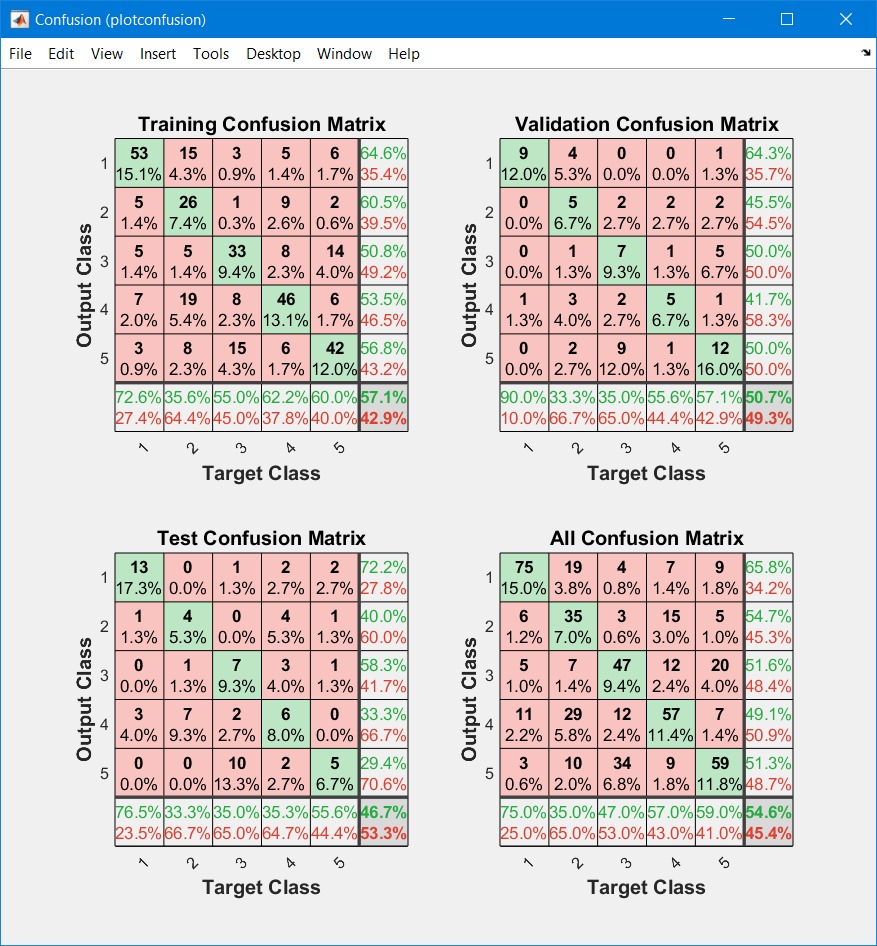
Out of 25 cycles the best performance is obtained for the dataset is observed at epoch 19.

We obtained the plot for performance of training set, validation set and test set for Class 1 to Class 5 of alphabets.





Given below is the plot for the 24 feature matrix i.e. 24 boxes considering the Output class and Target class.



**Conclusion** –

I was able to successfully run the code for Printed Alphabet Recognition on MATLAB with Classification of all 5 alphabet based on the concept of Neural Network method back propagation. First we collected the image dataset of 500 images. We used 15 % for test set, 15% for validation set and 70% for training data Resized the images into 24 pixels Black-white on Matlab with command imresize (imread) , we observed that the shaded region with alphabet was considered as 0 rest as 1 that is how binary function for 0 and 1. We tested the deep learning neural network with selecting input nodes, output nodes, weight matrices, who data are passed into the network and how to calculate. Overall we got a good amount of knowledge on what machine learning really is and how beneficial it is for Pattern recognition.

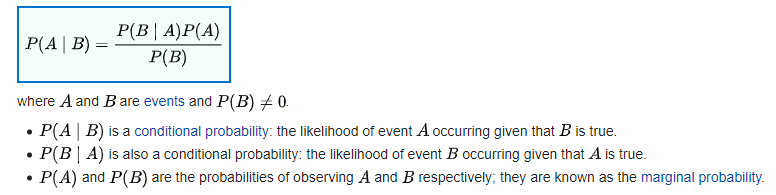
**Experiment No : 2**

**Title:** Naïve Bayes for D & E Alphabet Classification

**Theory:**

I used Naïve Bayes Classification for multiclass learning here 3 and 2 classes respectively. Naïve Bayes classifiers store the training data, parameter values, data distribution, and prior probabilities

**Bayes Theorem :** Bayes' theorem describes the probability of an event based on prior knowledge (supervised) of conditions that might be related to the event.



**Posterior Probability and Prior Probability -** A posterior probability is the probability of assigning observations to groups given the data like [P(A/B)]. A prior probability is the probability that an observation will fall into a group before you collect the data like [P(B)]

We Construct a Naive Bayes classifier for Fisher's iris & alphabet data

**Dataset used :**

Type of dataset (Isis, Alphabet)

For the second dataset(Alphabet csv 24 features files) we use character ‘d’ and ‘e at probability 0.50 each

Normalization have been performed of all the dataset i.e all values lie between 0-1.

**Characters in training dataset =** 50 samples

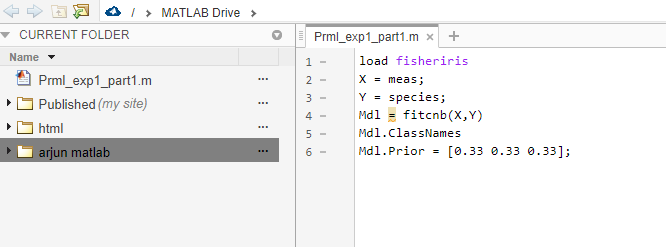
**Characters in testing dataset =** 50 samples

**Nr. of features used for character detection =** 24

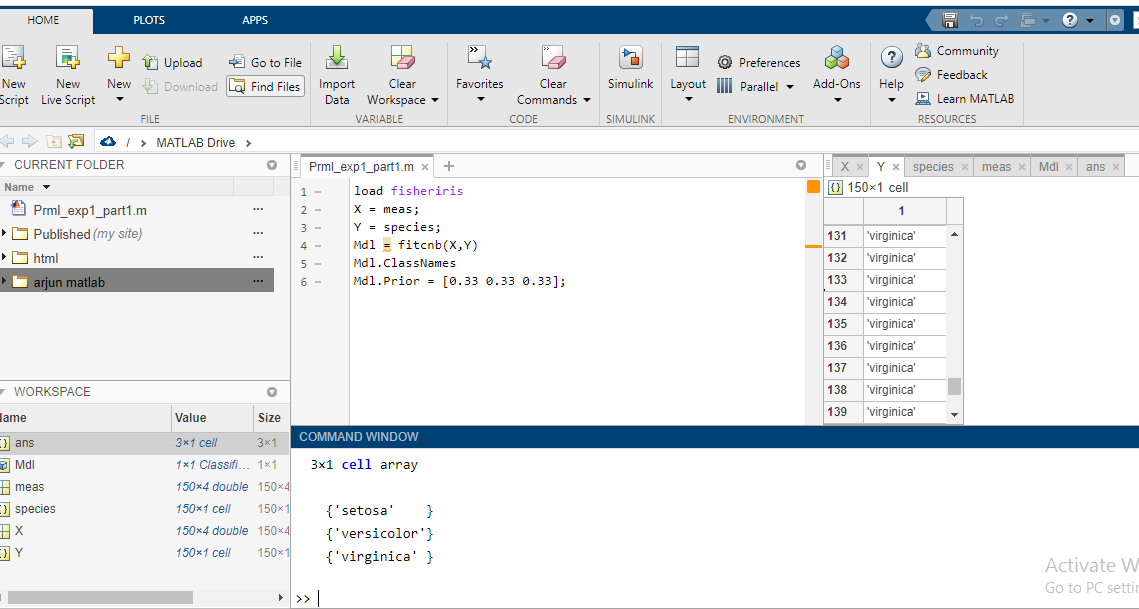
Total 100 datapoints

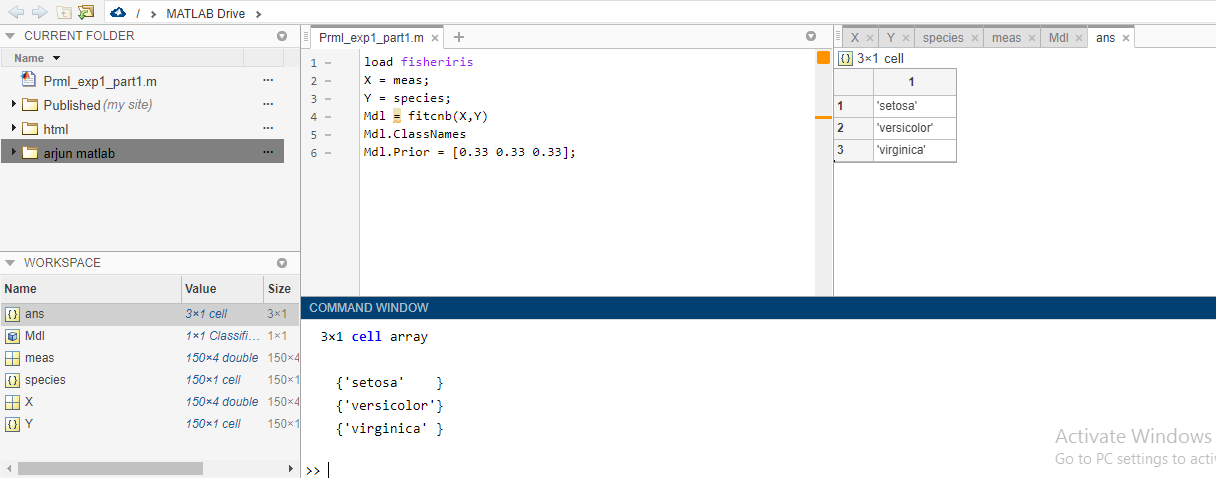
We have 2 outputs i.e. class 0 and class 1

**CODE NR. 1 :**

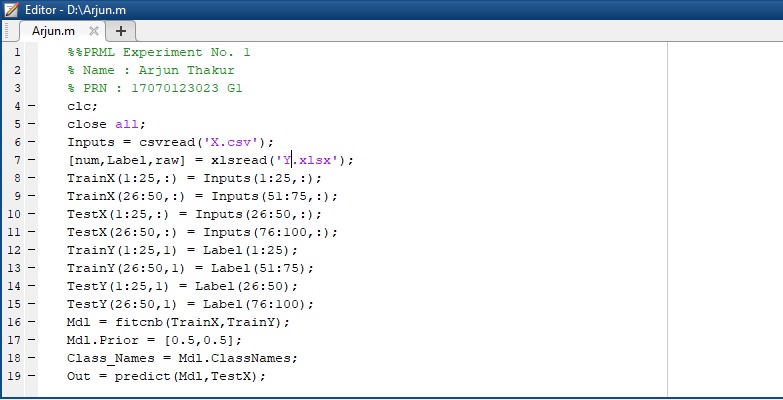


**OUTPUT NR. 1 :**

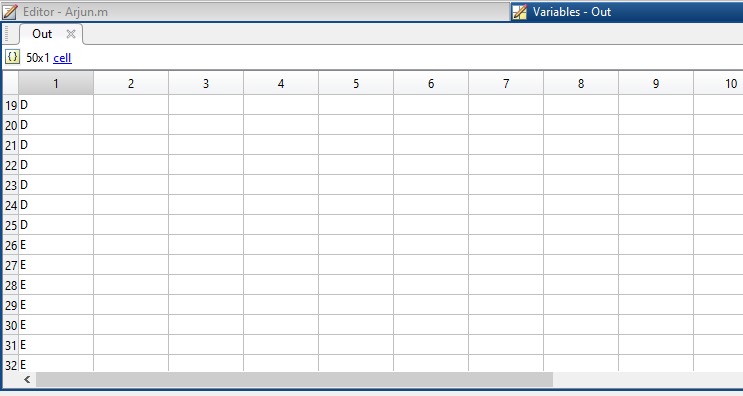




**CODE NR. 2 :**



**OUTPUT NR.2 :**



**Conclusion :** I got introduced to Bayes Theorem and how it is useful in Machine Learning. On change of prior probability the output y dataset ratio of both classification gets changed. I got a chance to work on Alphabet dataset and got introduced to CSV files too.

**Experiment No : 3**

**Aim:** To build up K-means & Hierarchical Clustering Models.

**Theory:**

**Clustering: Clustering** is a Machine Learning technique that involves the grouping of data points. Data points that are in the same group should have similar properties and/or features, while data points in different groups should have highly dissimilar properties and/or features.

**K-means Clustering**- K-means clustering is a type of unsupervised learning, which is used when you have unlabelled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K.

**Hierarchical Clustering -** Hierarchical clustering is a powerful technique that allows you to build tree structures from data similarities.

I have used Statistics and Machine Learning Toolbox for performing Clustering.

**Code 1: K-means Clustering**

clc; clear all; close all;

k = csvread('CSV\_Features\_24.csv');

knew = zeros(150,24);

knew(1:50,:) = k(1:50,:);

knew(51:100,:) = k(51:100,:);

knew(101:150,:) = k(101:150,:);

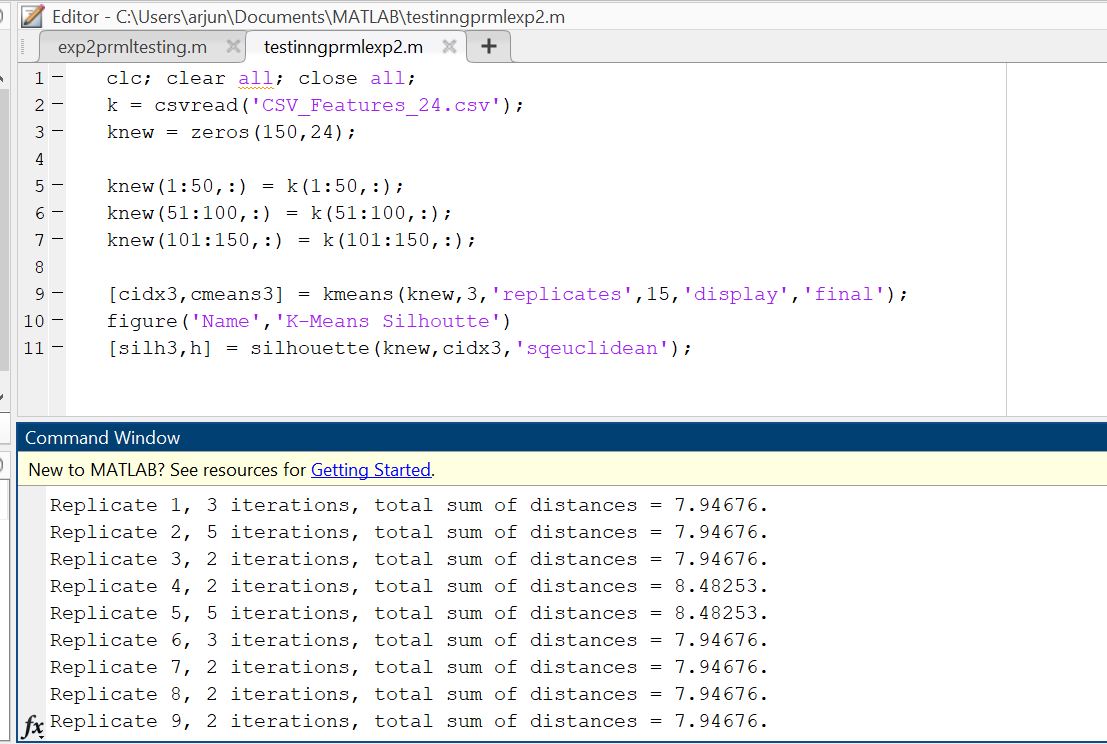
[cidx3,cmeans3] = kmeans(knew,3,'replicates',15,'display','final');

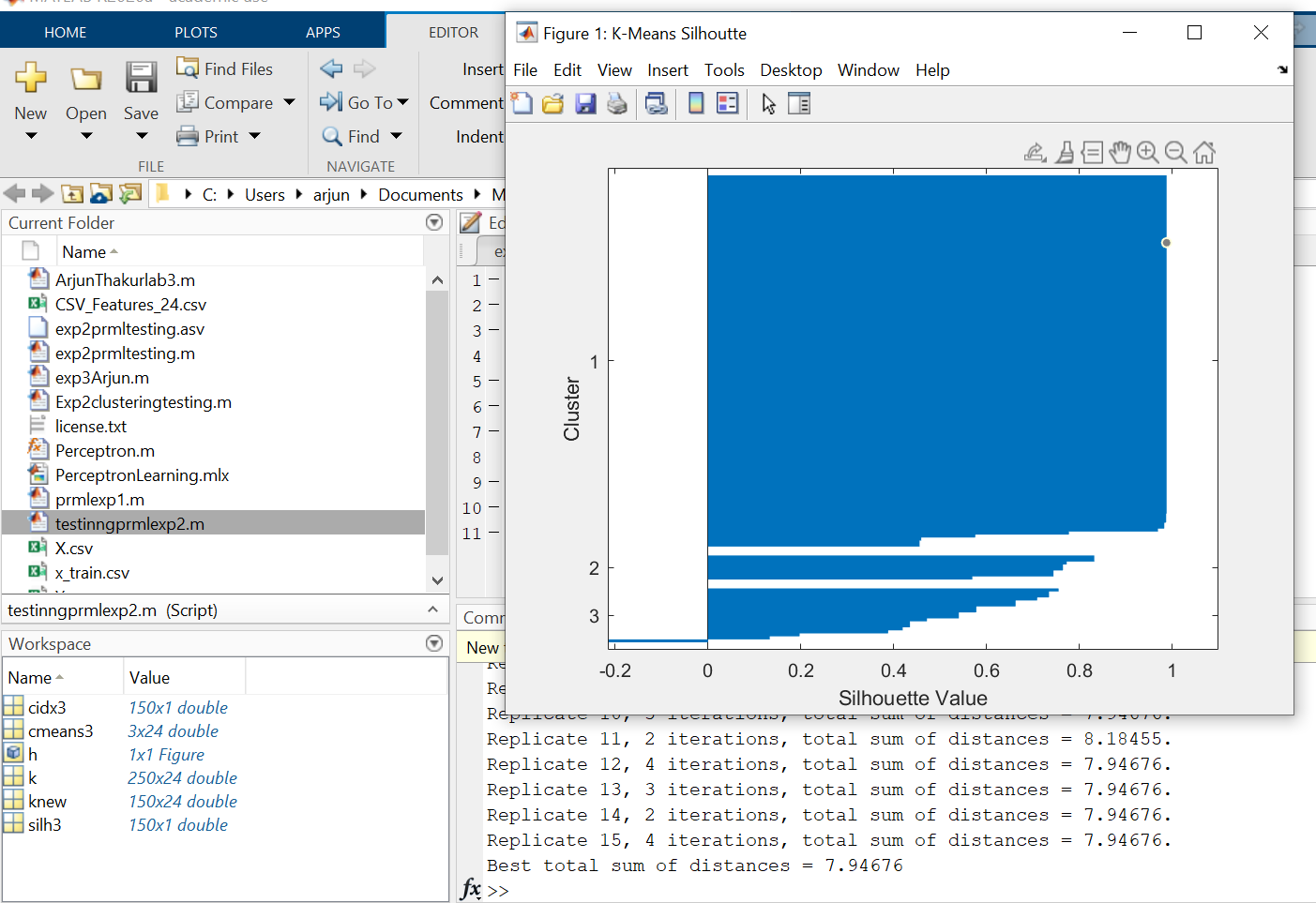
figure('Name','K-Means Silhoutte')

[silh3,h] = silhouette(knew,cidx3,'sqeuclidean');

**Dataset used :** CSV files with clustering feature data

**Output 1 :**





**Code 2: Hierarchical Clustering**

k = csvread('CSV\_Features\_24.csv');

knew = zeros(150,24);

knew(1:50,:) = k(1:50,:);

knew(51:100,:) = k(51:100,:);

knew(101:150,:) = k(101:150,:);

X = pdist(knew,'euclidean');

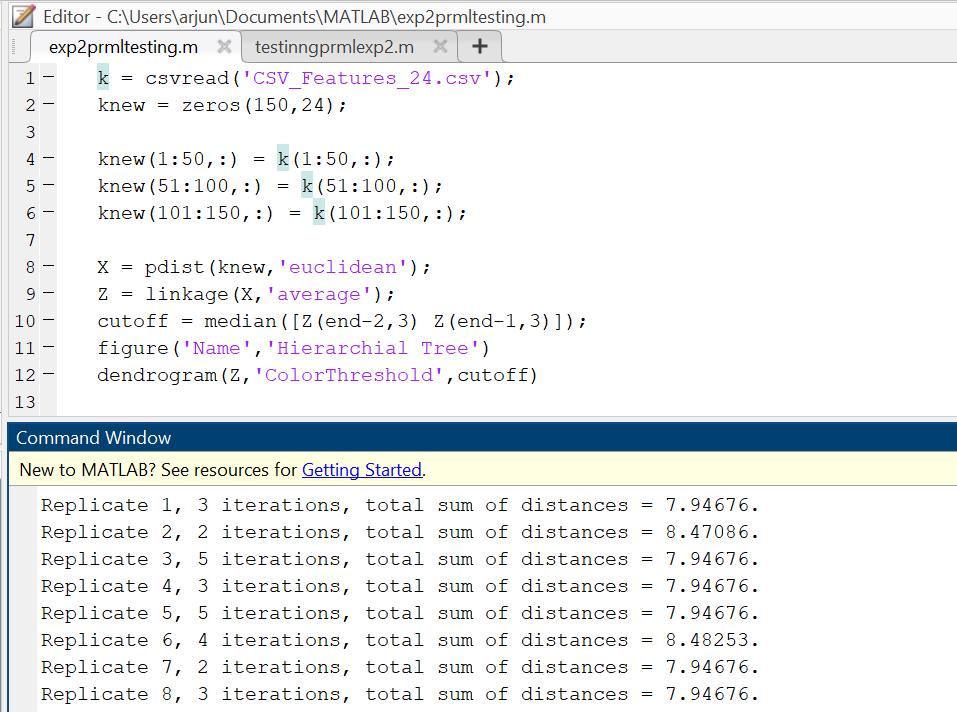
Z = linkage(X,'average');

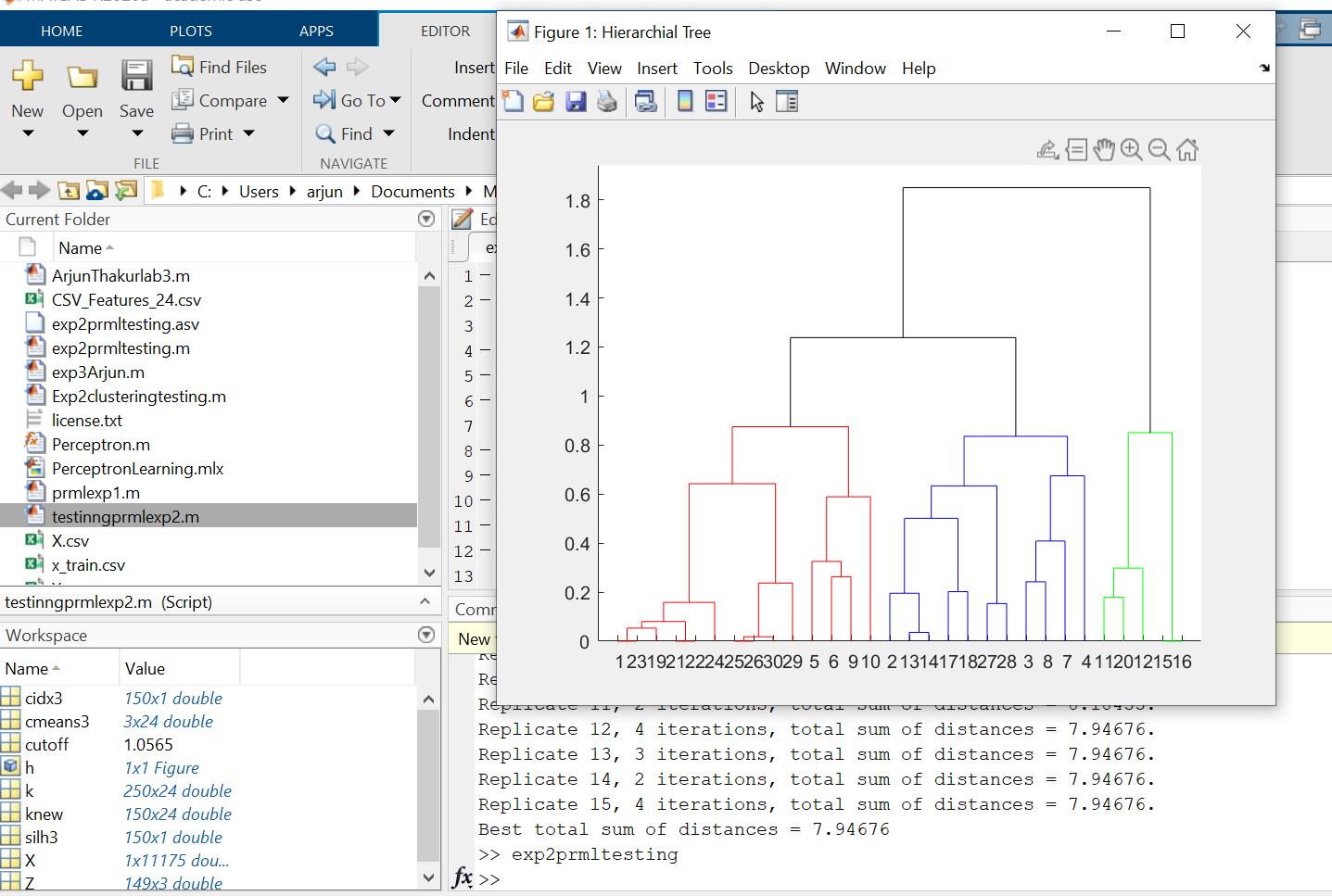
cutoff = median([Z(end-2,3) Z(end-1,3)]);

figure('Name','Hierarchial Tree')

dendrogram(Z,'ColorThreshold',cutoff)

**Output 2:**





**Conclusion** : Therefore, I have understood about unsupervised learning (Clustering) in more depth to perform K- means Clustering and Hierarchical Clustering on MATLAB software.

**Experiment No : 4**

**Aim :** Implementing AND Gate and OR Gate using Perceptron

**Theory :** Perceptrons are simple single-layer binary classifiers, which divide the input space with a linear decision boundary. Perceptrons can learn to solve a narrow range of classification problems like Logic Gates. The perceptron is an algorithm for [supervised learning](https://en.wikipedia.org/wiki/Supervised_classification) of [binary classifiers](https://en.wikipedia.org/wiki/Binary_classification). A binary classifier is a function which can decide whether or not an input, represented by a vector of numbers, belongs to some specific class.[[1]](https://en.wikipedia.org/wiki/Perceptron#cite_note-largemargin-1) It is a type of [linear classifier](https://en.wikipedia.org/wiki/Linear_classifier), i.e. a classification algorithm that makes its predictions based on a [linear predictor function](https://en.wikipedia.org/wiki/Linear_predictor_function) combining a set of weights with the [feature vector](https://en.wikipedia.org/wiki/Feature_vector).

**Code 1 :**

**AND gate (AND GATE)**

clear all;clc;

Input=[-1 -1;-1 1;1 -1;1 1];

y=[-1 -1 -1 1];

Initial\_weights=[-1 -1]';

eta=0.7;

weights= Perceptron(Input,y,Initial\_weights,eta)

x\_test = [1 -1];

y\_in = sum(x\_test.\* weights);

if y\_in > 0

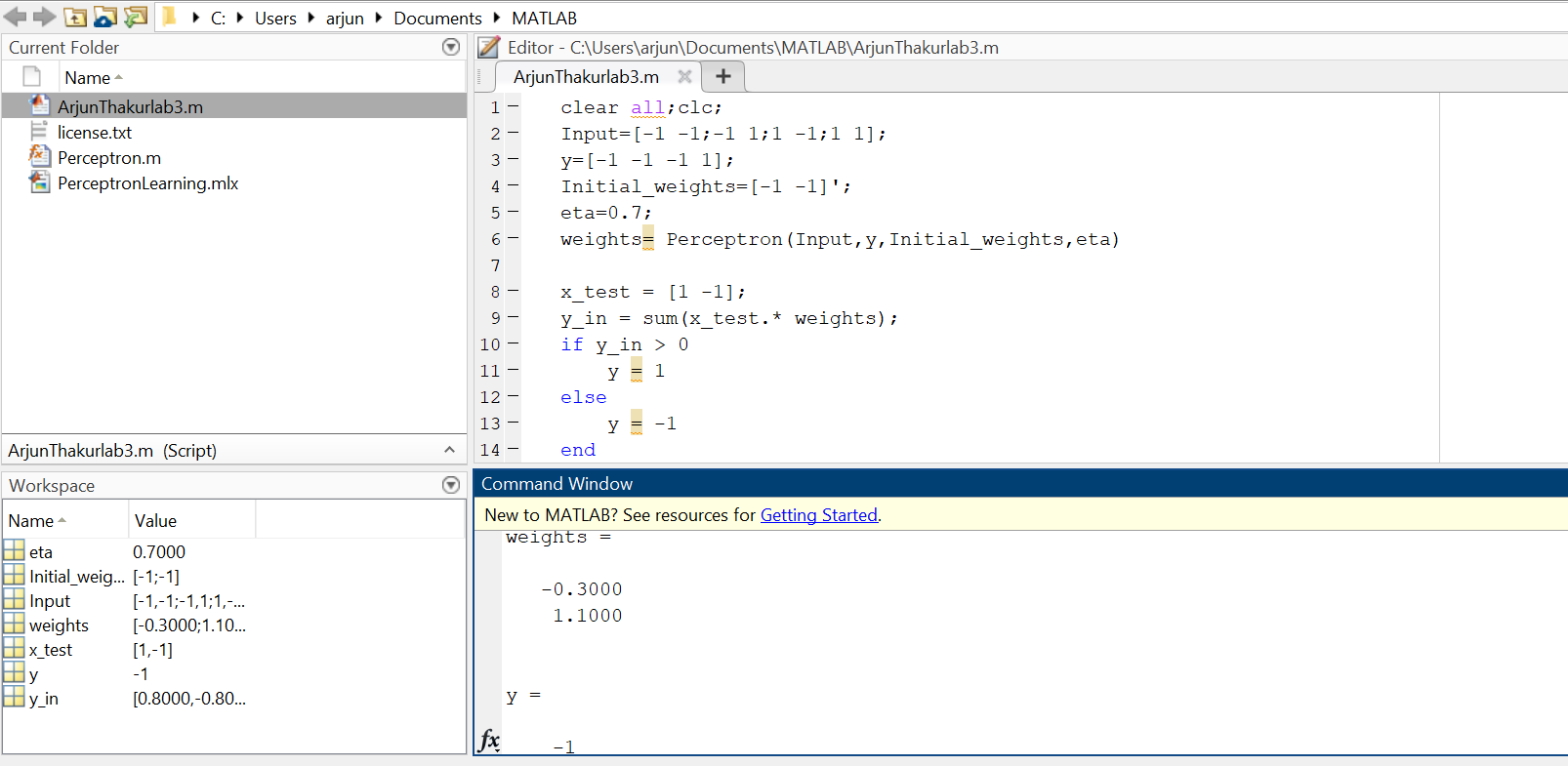
y = 1

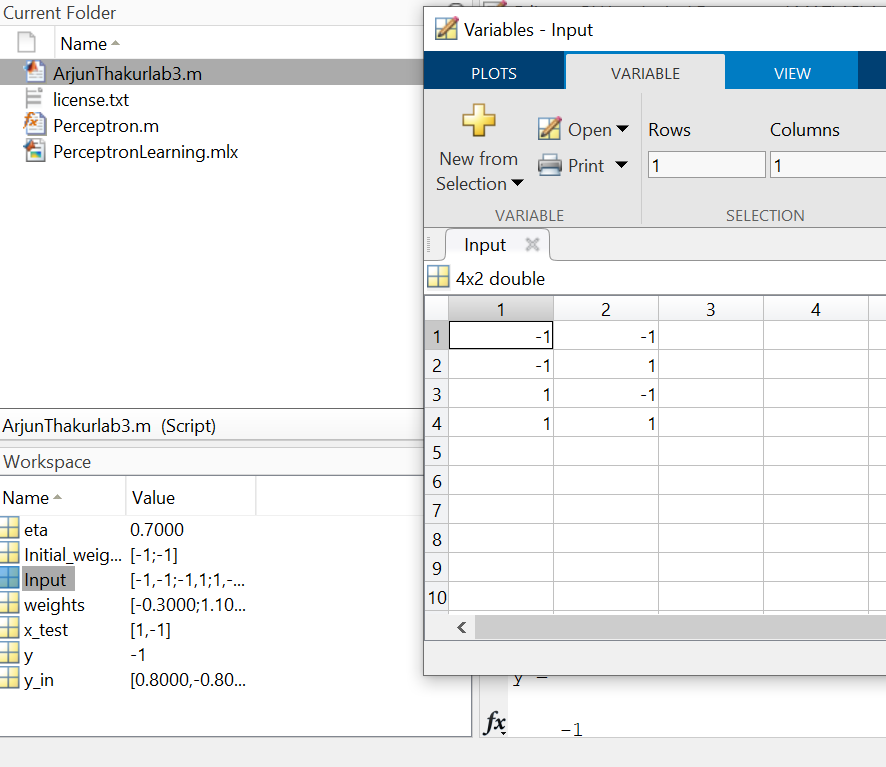
else

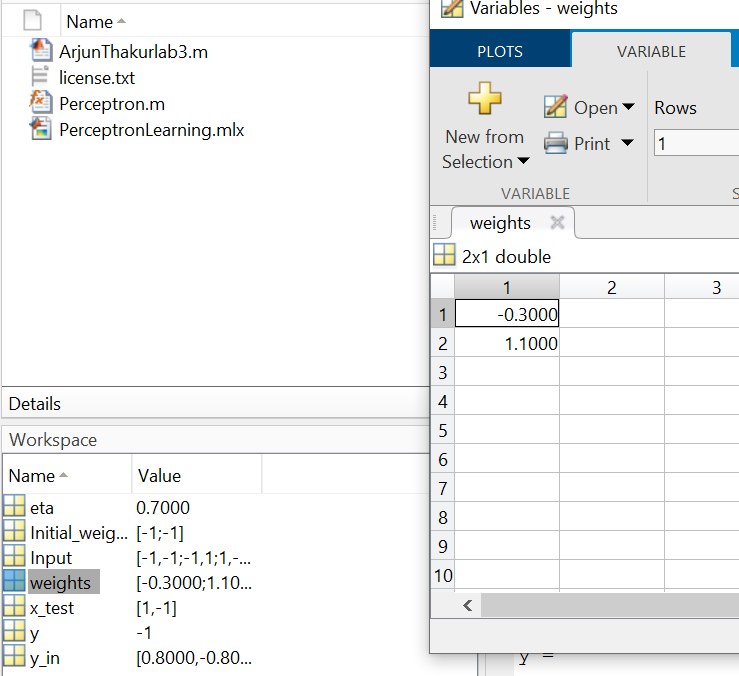
y = -1

end

**Output 1:**







**Code 2 (OR GATE) :**

clear all;clc;

Input=[-1 -1;-1 1;1 -1;1 1];

y=[-1 1 1 1];

Initial\_weights=[-1 -1]';

eta=0.7;

weights= Perceptron(Input,y,Initial\_weights,eta)

x\_test = [1 -1];

y\_in = sum(x\_test.\* weights);

if y\_in > 0

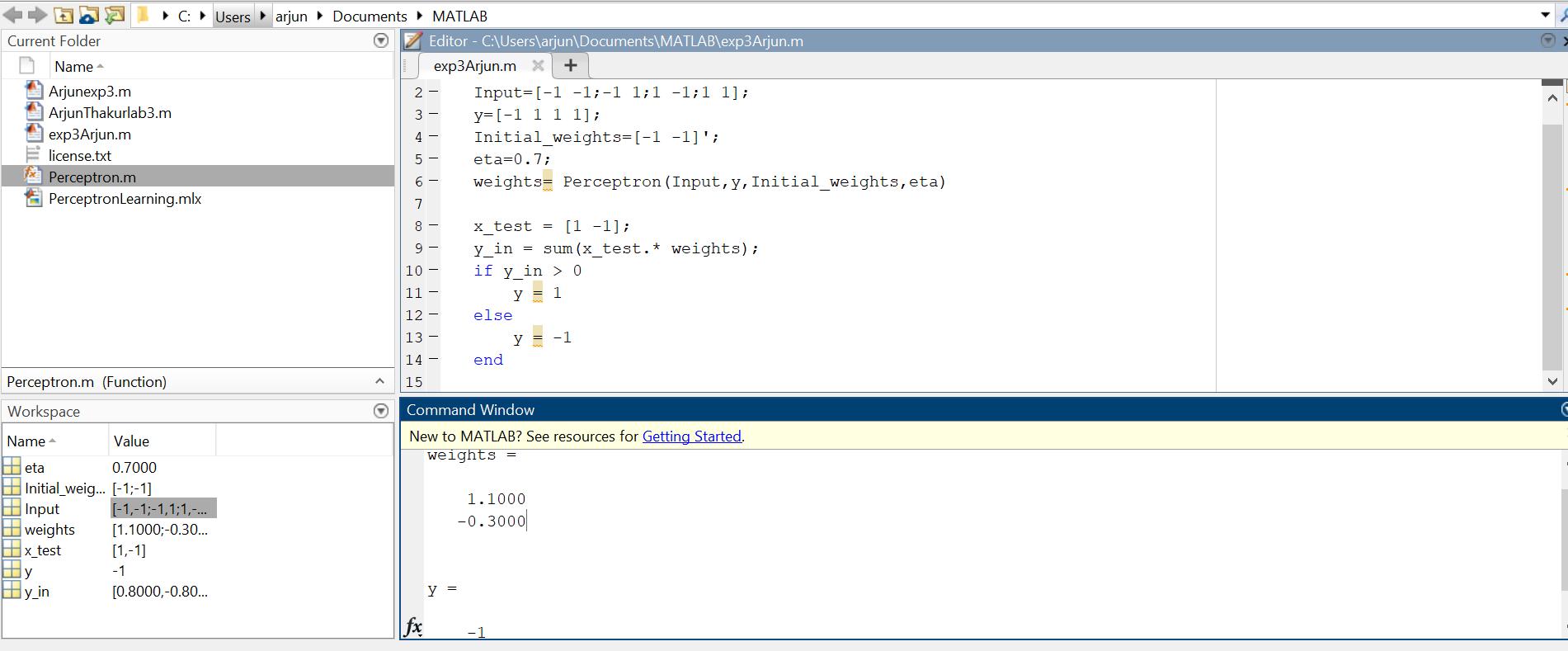
y = 1

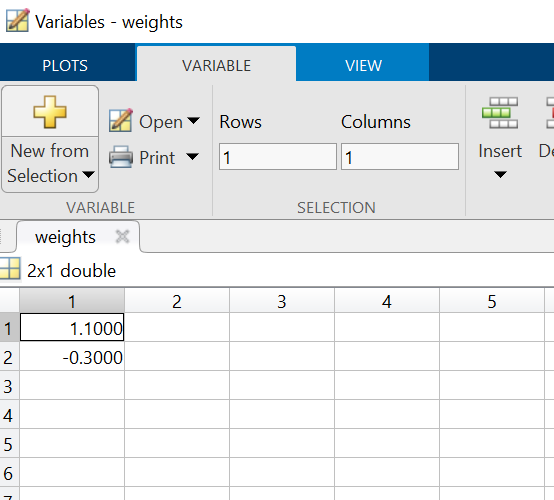
else

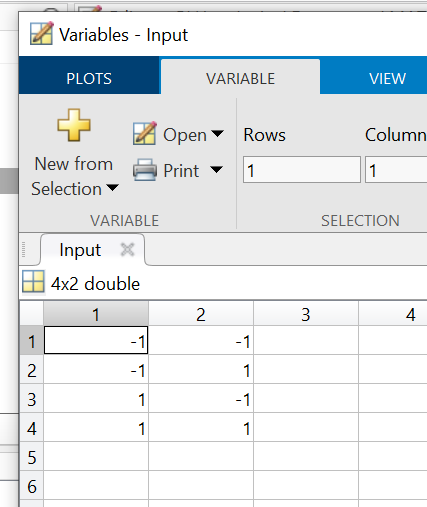
y = -1

end

**Output 2:**







**Conclusion :** Therefore I performed the experiment for AND &OR gate perceptron using the perceptron function from Mathworks and observed the final weights in the workspace. I also saw that threshold, activation function were performed itself with the help of perceptron function.

**Experiment No. 5**

**Aim :** Implementation of EX-OR gate using Backpropagation algorithm

**Theory :**

**Backpropagation :** In [machine learning](https://en.wikipedia.org/wiki/Machine_learning), backpropagation (backprop, BP) is a widely used [algorithm](https://en.wikipedia.org/wiki/Algorithm) in training [feedforward neural networks](https://en.wikipedia.org/wiki/Feedforward_neural_network) for [supervised learning](https://en.wikipedia.org/wiki/Supervised_learning). Generalizations of backpropagation exist for other [artificial neural networks](https://en.wikipedia.org/wiki/Artificial_neural_network) (ANNs), and for functions generally – a class of algorithms referred to generically as "backpropagation". In [fitting a neural network](https://en.wikipedia.org/wiki/Artificial_neural_network#Learning), backpropagation computes the [gradient](https://en.wikipedia.org/wiki/Gradient) of the [loss function](https://en.wikipedia.org/wiki/Loss_function) with respect to the [weights](https://en.wikipedia.org/wiki/Glossary_of_graph_theory_terms#weight) of the network for a single input–output

**Functions used:**

1. X = net(p,n) returns the first n points from the point set p, which is either a haltonset or sobolset object. X is an n-by-d matrix, where d is the number of dimensions of the points in p. The object p encapsulates properties of a specified quasirandom sequence. Values of the point set are generated whenever you access p using net or parenthesis indexing. Values are not stored within p.

2. BP\_TB: this function allows us to train a multi-layer perceptron based on back propagation of the gradient "for regression".

**Code :**

%Experiment 4 Backpropagation using EX-OR gate

% Arjun Thakur 17070123023

clear all ;

clc;

%% Loading Truth Table for EX-OR gate

x=[0 0;0 1;1 0;1 1]';

y=[0 1 1 0]';

%% Initialize parameters

desired\_error=1e-2;

Learning\_Rate=0.9;

hidden\_layers=[1];

plotting='yes';

%% Training

[net]=BP\_TB(x,y,desired\_error,Learning\_Rate,hidden\_layers,plotting);

%%%%%%%%%%% prediction

%% Prediction

[outputs]=predict(net,x);

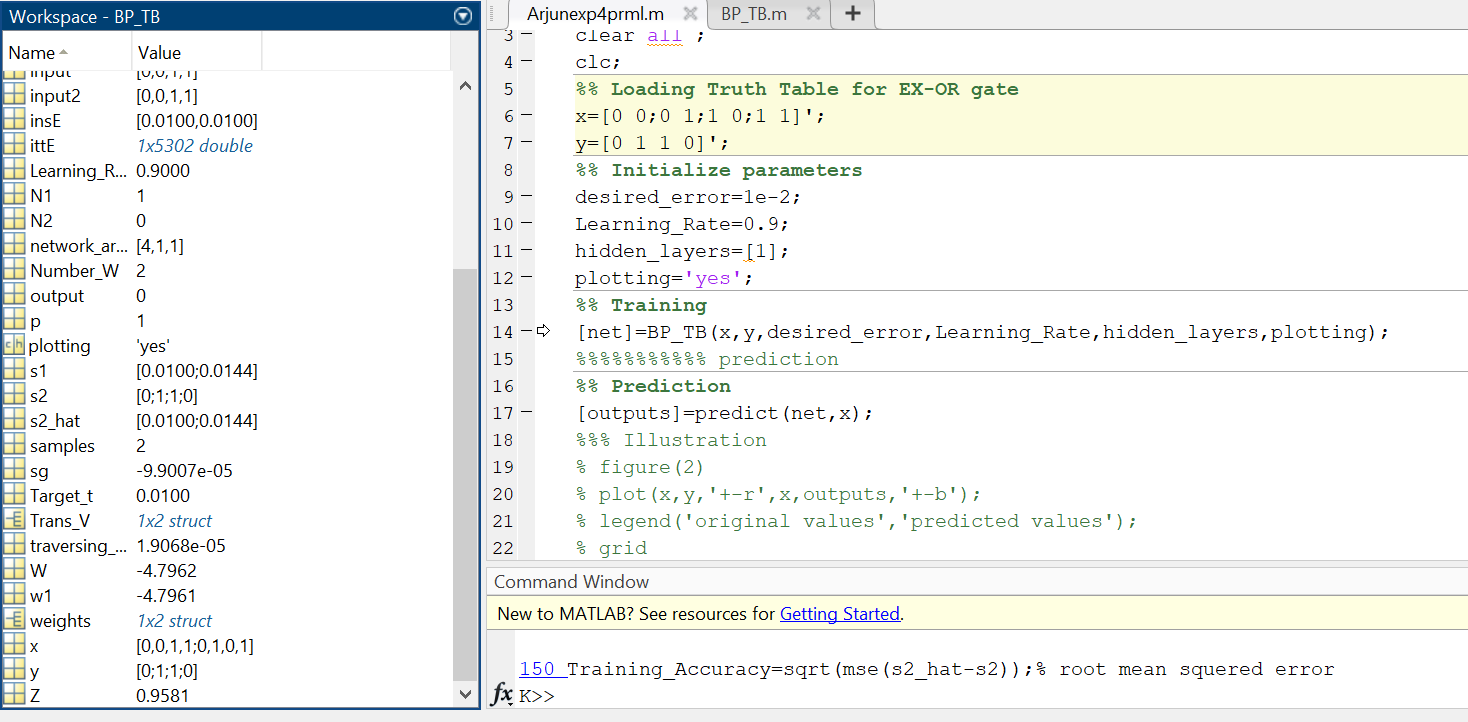
%%% Illustration

% figure(2)

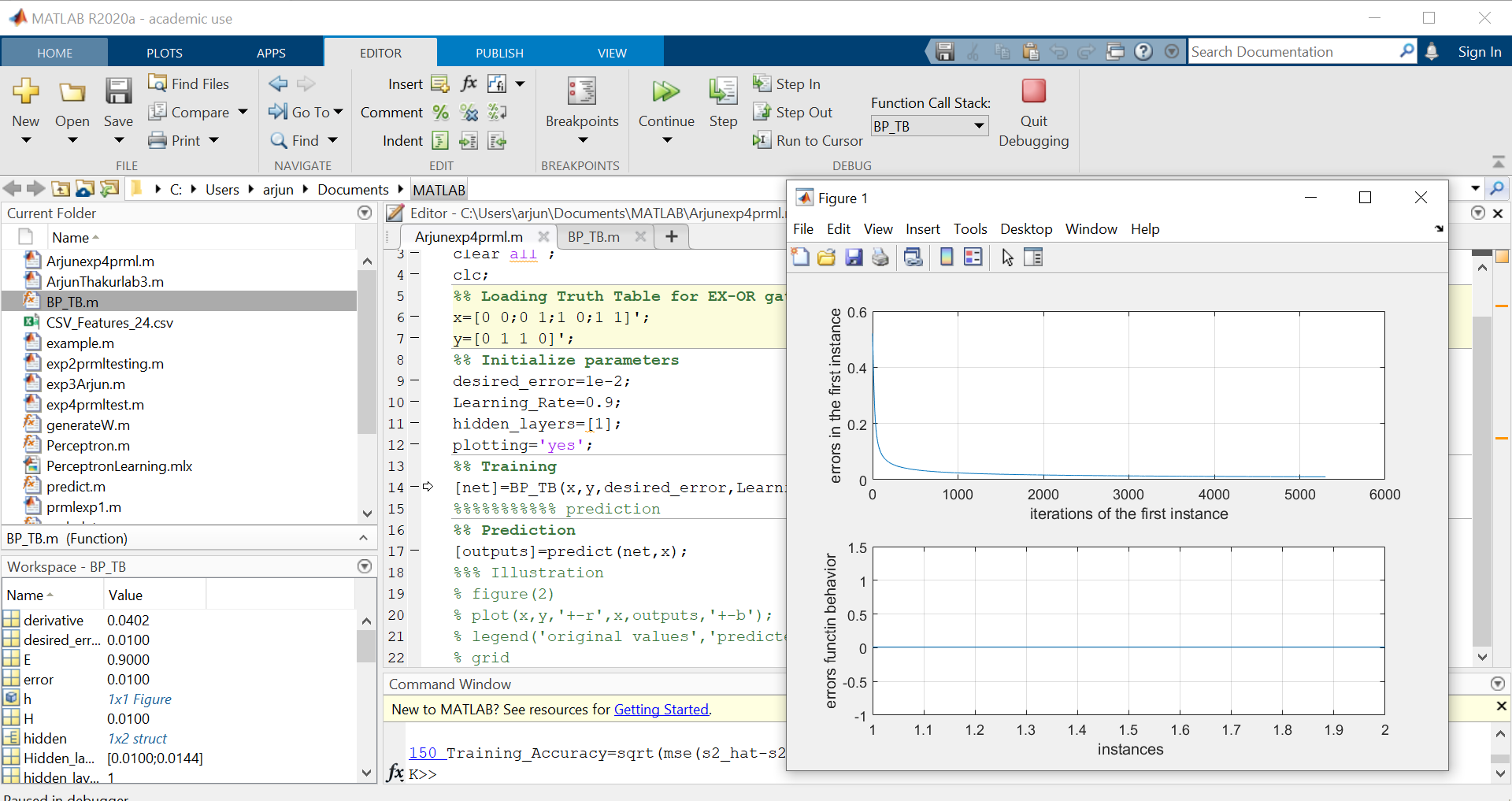
% plot(x,y,'+-r',x,outputs,'+-b');

% legend('original values','predicted values');

% grid



**Output :**



**Conclusion :** Thus, I was able to perform Backpropagation for supervised learning using 1 hidden layer for EX-OR Gate and observe the plot for error in first instance and error function behaviour wrt iterations of 1st instance and instances.

**Experiment No : 6**

**Aim:** Classification of A & B English letters using backpropagation

**Theory:** Backpropagation is a short form for "backward propagation of errors." It is a standard method of training artificial neural networks. This method helps to calculate the gradient of a loss function with respects to all the weights in the network. The Backpropagation neural network is a multilayer, feedforward neural network and is by far the most extensively used.

**Dataset used :** CSV files with A & B alphabet 24 features data

**Code:**

clc;

clear all;

close all;

X = transpose(csvread('Prml\_exp5(X).csv')); %Input 24 features of A

t = transpose(csvread('Prml\_exp5(Y).csv')); %Alphabet a and b character as target

net = feedforwardnet(2,'traingd'); % 3 defines the input out and hidden layer

net.trainParam.show =50;

net.trainParam.lr = 0.05;

net.trainParam.epochs = 300;

net.trainParam.goal = 1e-5;

[net,tr] = train(net,X,t); %Training the network

view(net)

y = net(X);

testX = X(:,tr.testInd); % testing the network

testT = t(:,tr.testInd);

testY = net(testX);

YPredicted = net(X);

YPredicted(:,1:10)

figure(3)

plotconfusion(t,YPredicted)% plot the network training data

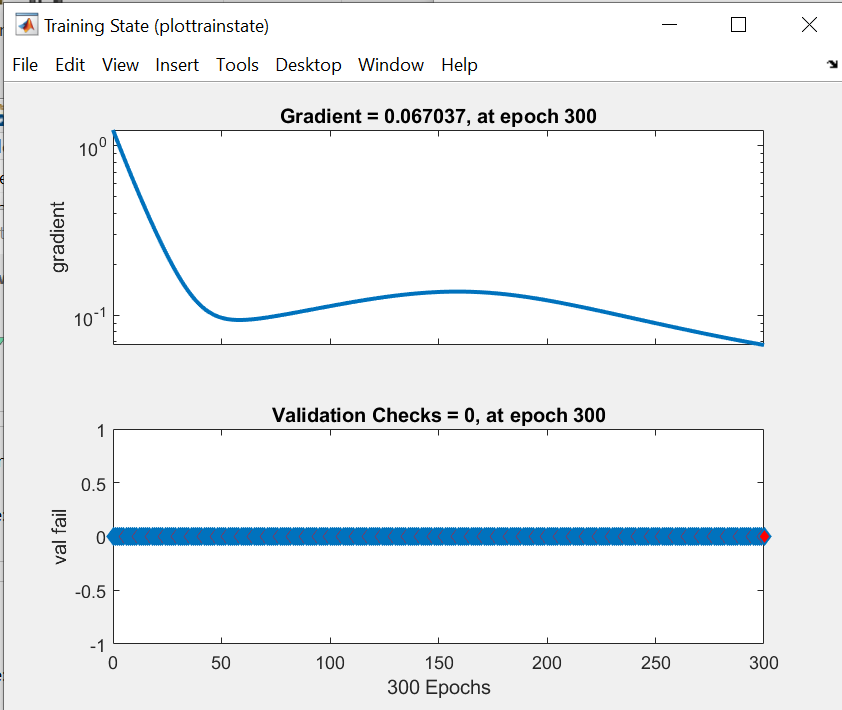
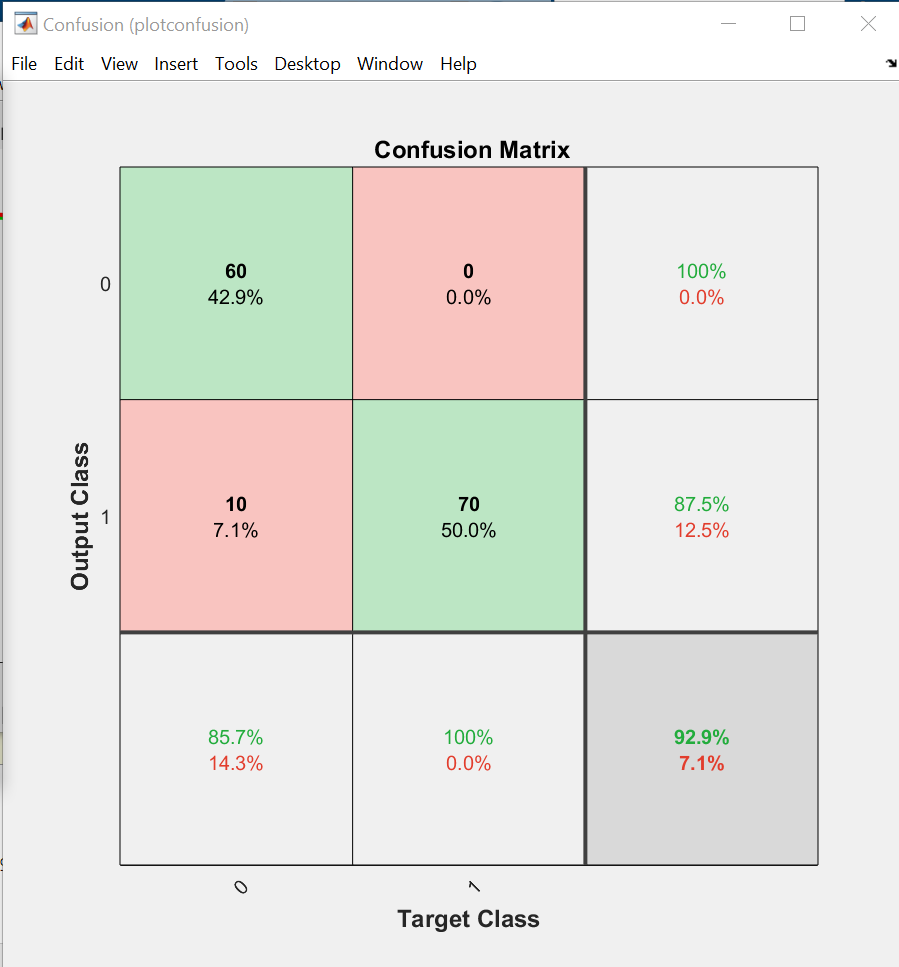
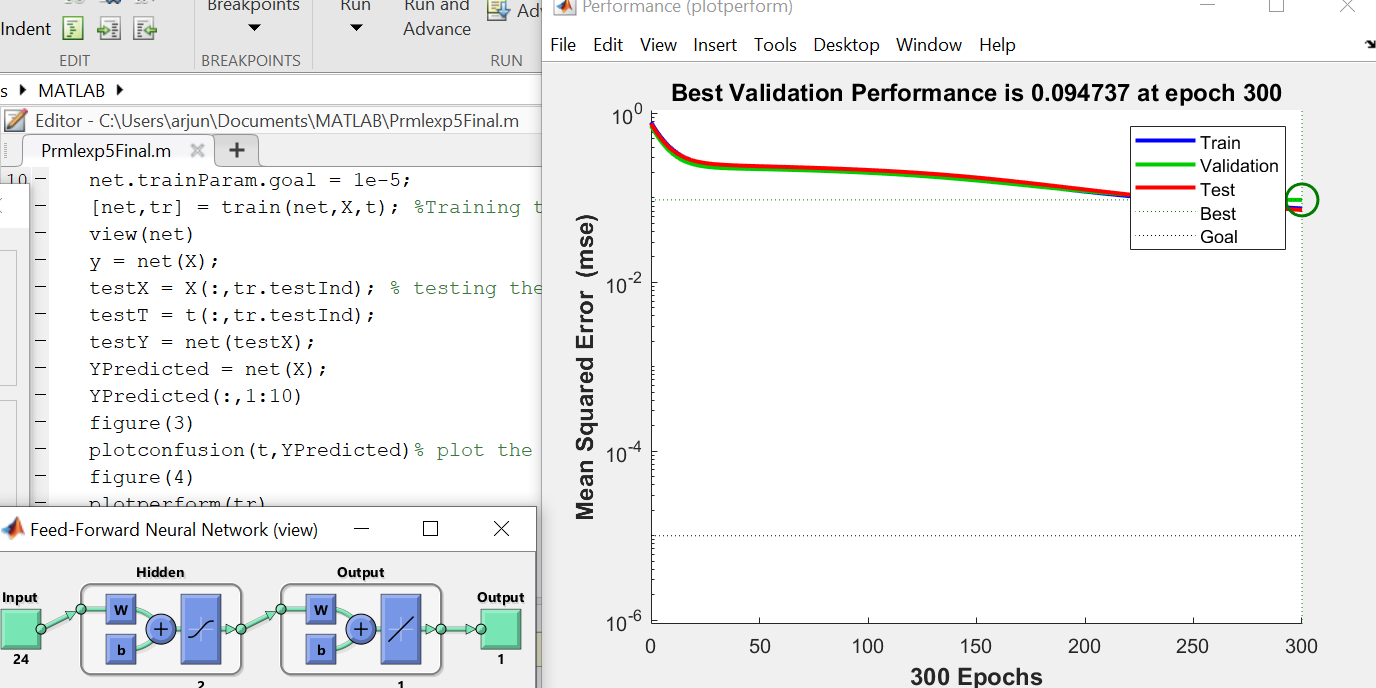
figure(4)

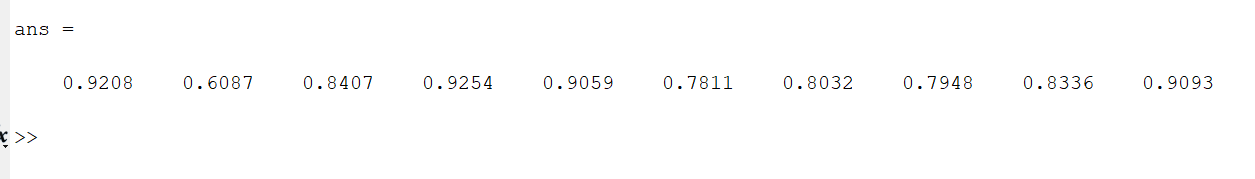
plotperform(tr)

figure(5)

plottrainstate(tr)

**Output-**





**Conclusion:**

I have successfully implemented letter classification using multi-layer backpropagation algorithm on MATLAB.

Experiment No. 7

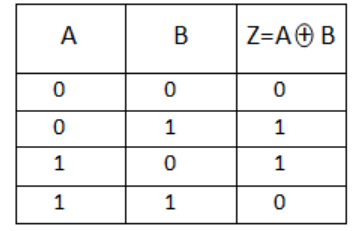
Aim : To implement SVM for EX-OR gate with 2 class (Class 0 and Class 1) & Fisher iris dataset (3D with 2 flowers)

**Theory :**

fitcsvm(): trains or cross-validates a support vector machine (SVM) model for one-class and two-class (binary) classification on a low-dimensional or moderate-dimensional predictor data set.

fitcsvm supports mapping the predictor data using kernel functions, and supports sequential minimal optimization (SMO), iterative single data algorithm (ISDA), or L1 soft-margin minimization via quadratic programming for objective-function minimization.  
Support Vector Machine (SVM) is a supervised classification model that helps in providing the most symmetrical decision boundary possible

1. **For EX-OR :**



1. **For Fisher Iris –** We have to Load Fisher's iris data set and remove the sepal lengths and widths and all observed setosa irises.

**Dataset used :** Fisher Iris dataset (in-built)

**Code 1 (EX-OR) :**

%%Experiment No. 6

% Arjun Thakur ENTC A

% 17070123023 (G1)

X=[0 0; 0 1; 1 0; 1 1];

Y=[0; 1; 1; 0];

Mdl=fitcsvm(X,Y,'KernelFunction','rbf');

sv=Mdl.SupportVectors;

figure

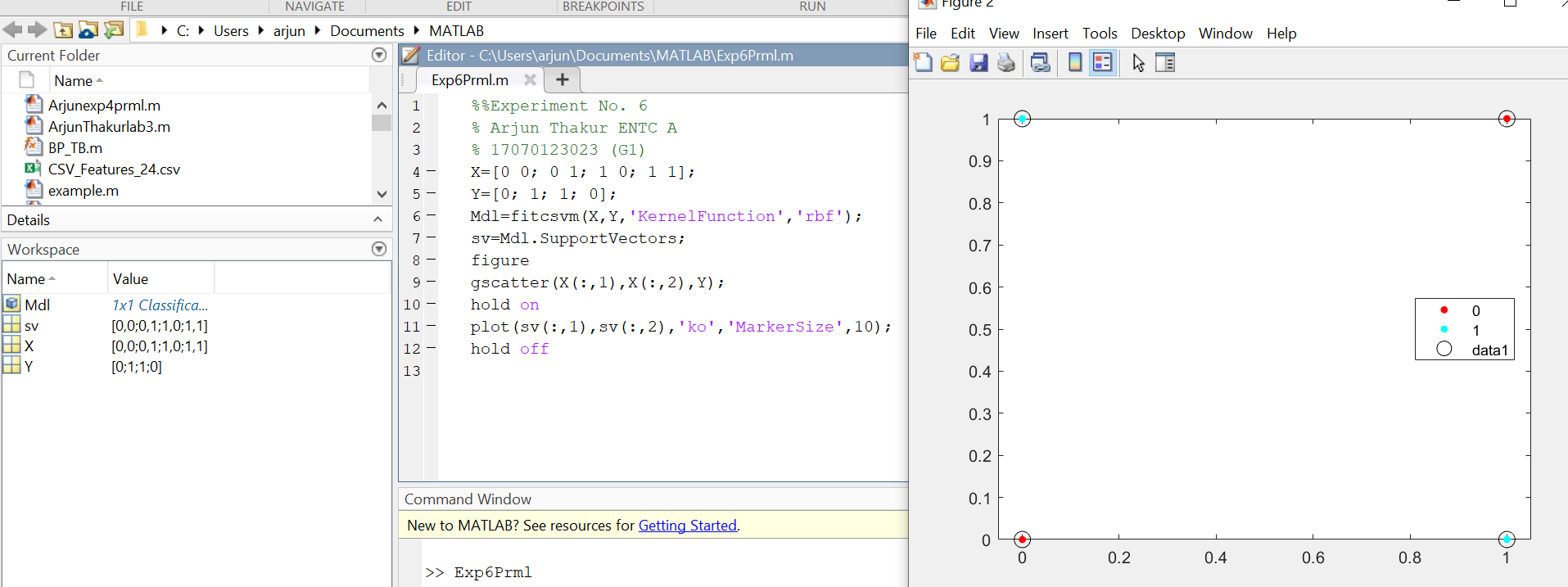
gscatter(X(:,1),X(:,2),Y);

hold on

plot(sv(:,1),sv(:,2),'ko','MarkerSize',10);

hold off

**Output :**



**Code 2 (Fisher Iris Dataset) :**

load fisheriris

inds = ~strcmp(species,'setosa');

X = meas(inds,3:4);

y = species(inds);

SVMModel = fitcsvm(X,y)

classOrder = SVMModel.ClassNames

sv = SVMModel.SupportVectors;

figure

gscatter(X(:,1),X(:,2),y)

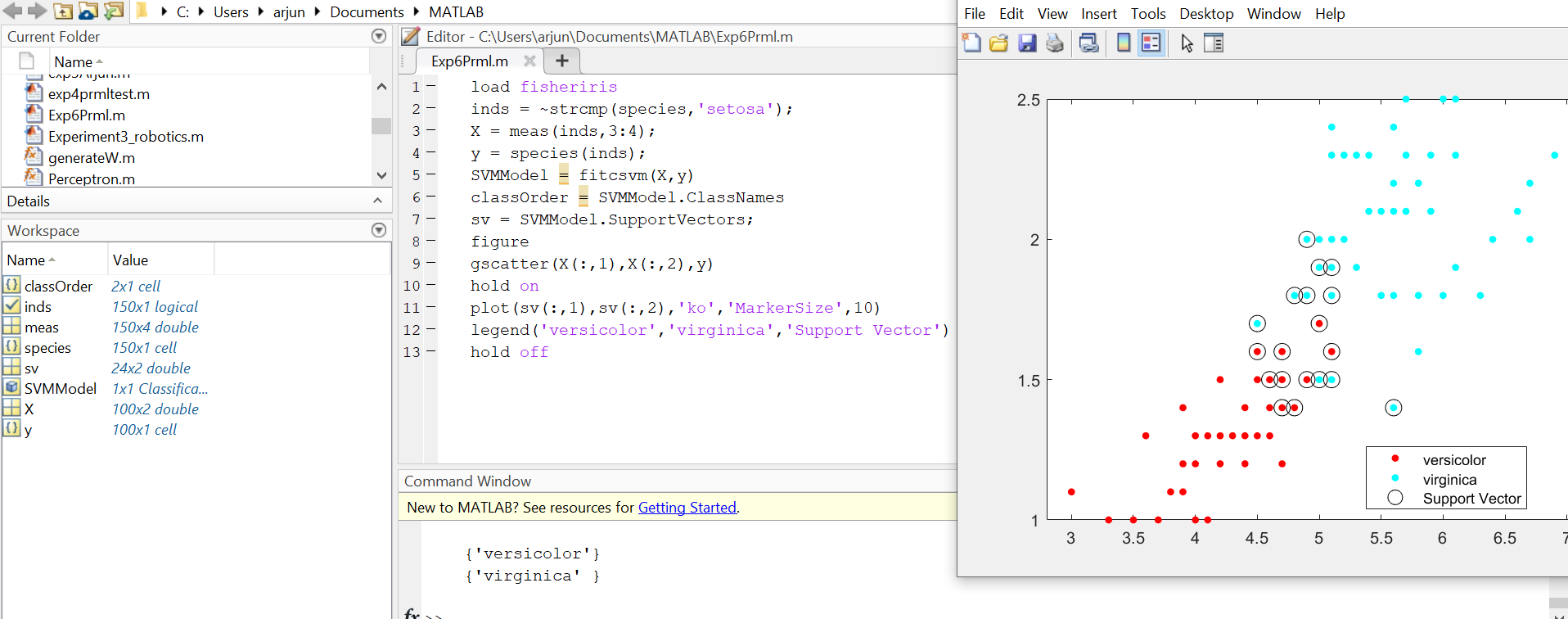
hold on

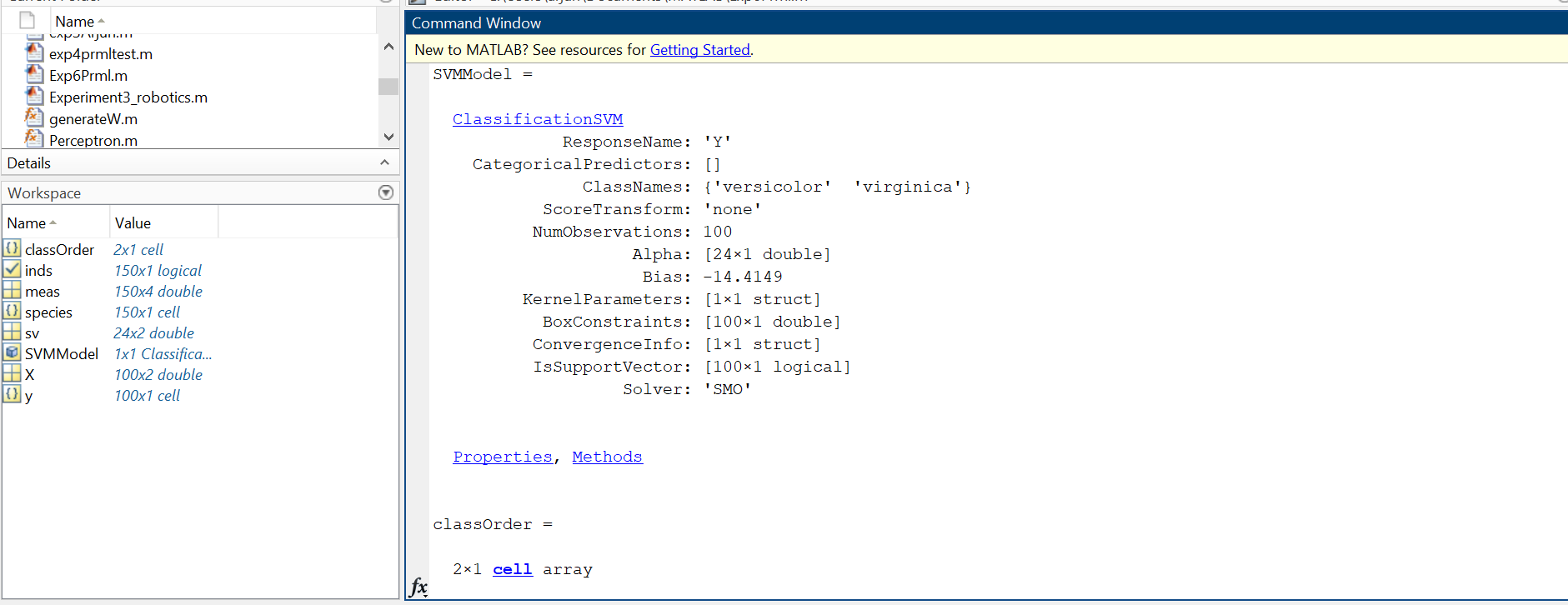
plot(sv(:,1),sv(:,2),'ko','MarkerSize',10)

legend('versicolor','virginica','Support Vector')

hold off

**Output & Command Window :**





## 

## Conclusion :

Thus I have learned how to perform SVM using EX-OR and Fisher iris dataset by picking up appropriate SVM function, Defining input, Kernel and ploting the data successfully on MATLAB*.* The support vectors are observations that occur on or beyond their estimated class boundaries. Thus, we can adjust the boundaries (and, therefore, the number of support vectors) by setting a box constraint during training using the 'BoxConstraint' name-value pair argument.

**Experiment No : 8**

**Aim :** To perform SVM classification on A & B Alphabets.

**Theory :**

SVM is a supervised machine learning algorithm which can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs.

I downloaded Image processing toolbox for this task.

**Dataset used :**

I have used 5 images of A and 5 images of B alphabets & an excel file with feature data of A & B Alphabets.

**Code :**

clc;

clear all;

rng(6,'twister')

D = dir('C:\Users\arjun\Documents\MATLAB\png\\*.png');

nfiles = length(D); %calculating the length of the dataset

for i = 1:nfiles

cfilename = D(i).name;

cimg = imread(cfilename);

I = imresize(cimg, [42 32]); %resizing the 'i'th image in the dataset to 42x32 pixels

threshold = graythresh(I); % calculating the threshold value for digitizing each image

I = imbinarize(I,threshold); %quatizing each image using the calculated threshold value

I = reshape(I,7,8,24); %reshaping the image to 24 boxes(features) with 7x8 pixels for each feature

I = mean(mean(I)); %calculating the mean for each featutre; mean(I) calculates the mean for 7x8 pixels for 24 boxes/features and mean(mean(I)) calculates the mean for values per box/feature

X(i,:)=I(:); %storing the 24 feature values of the image in the dataset matrix;

end

y= importdata('label.exp7.xlsx');

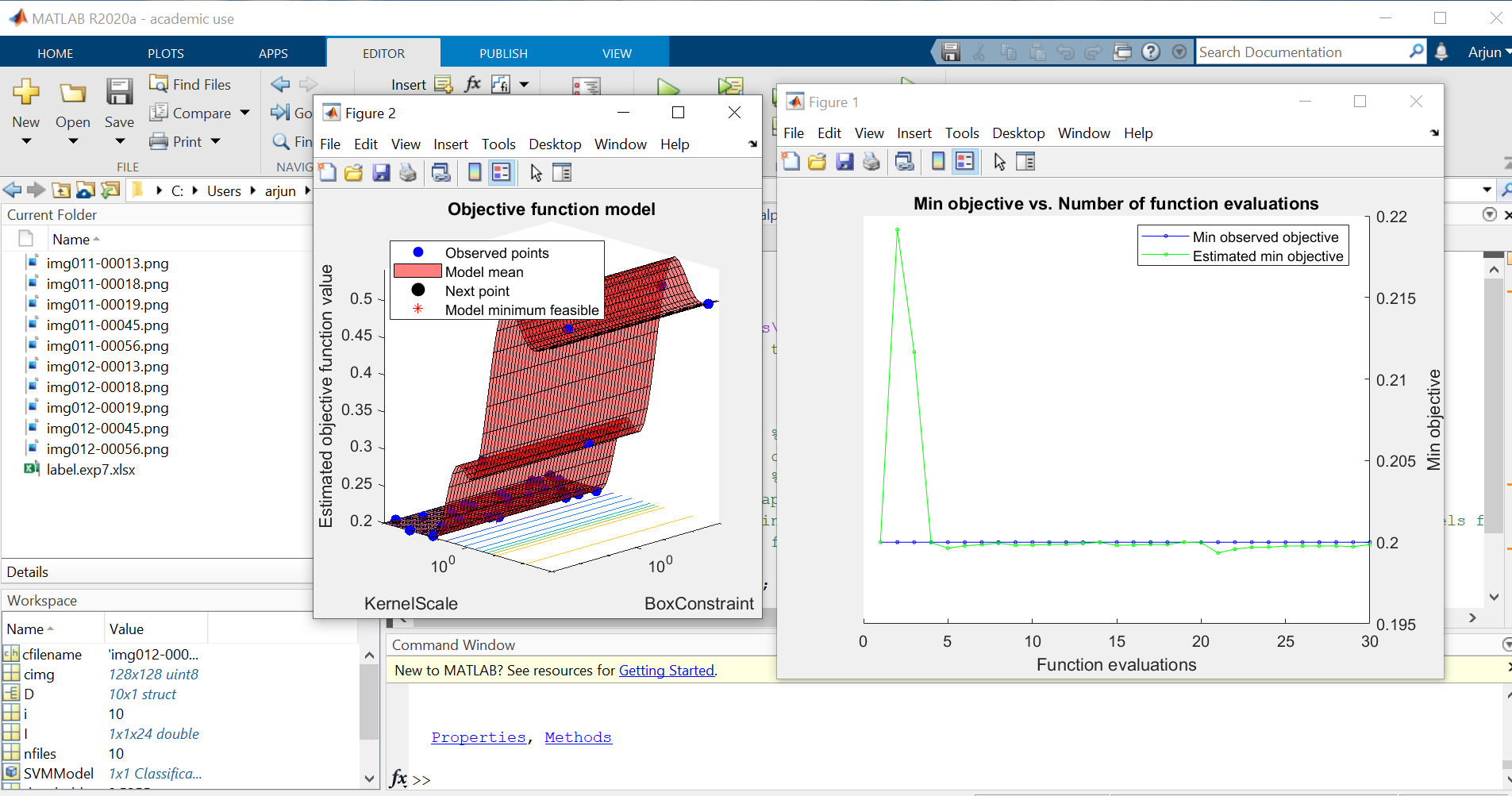
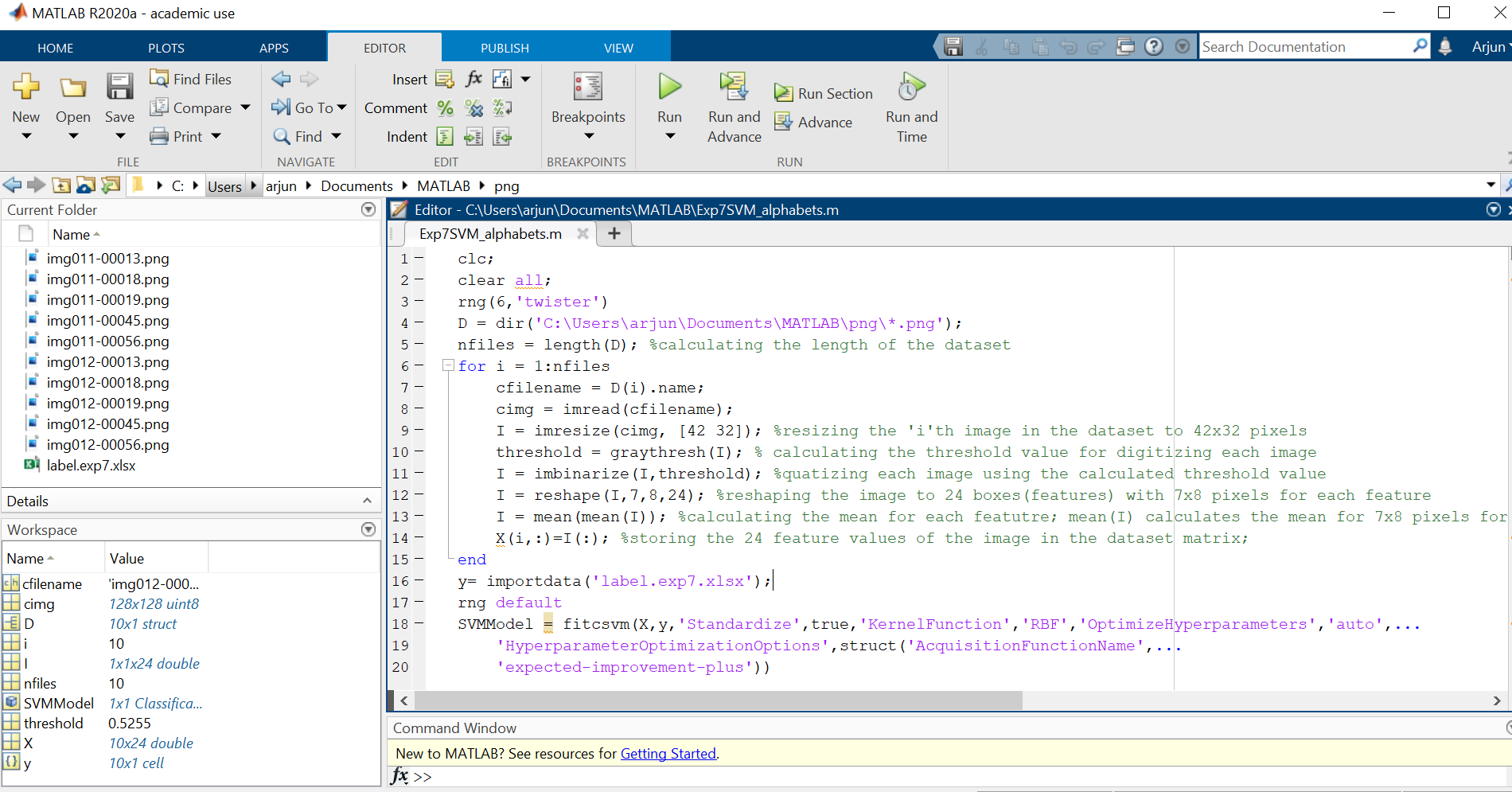
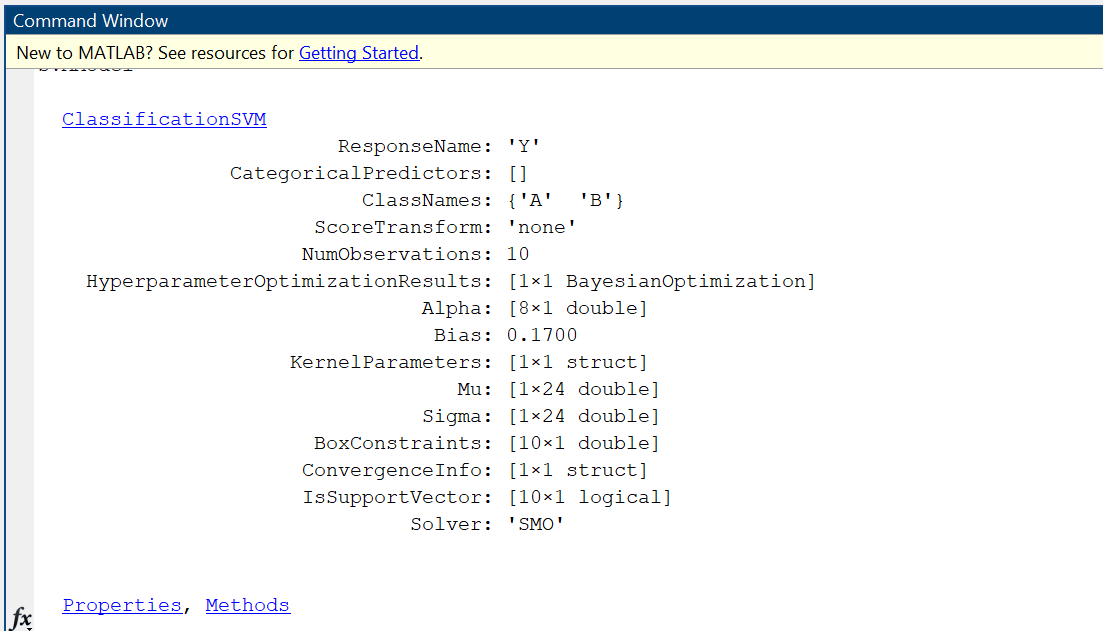
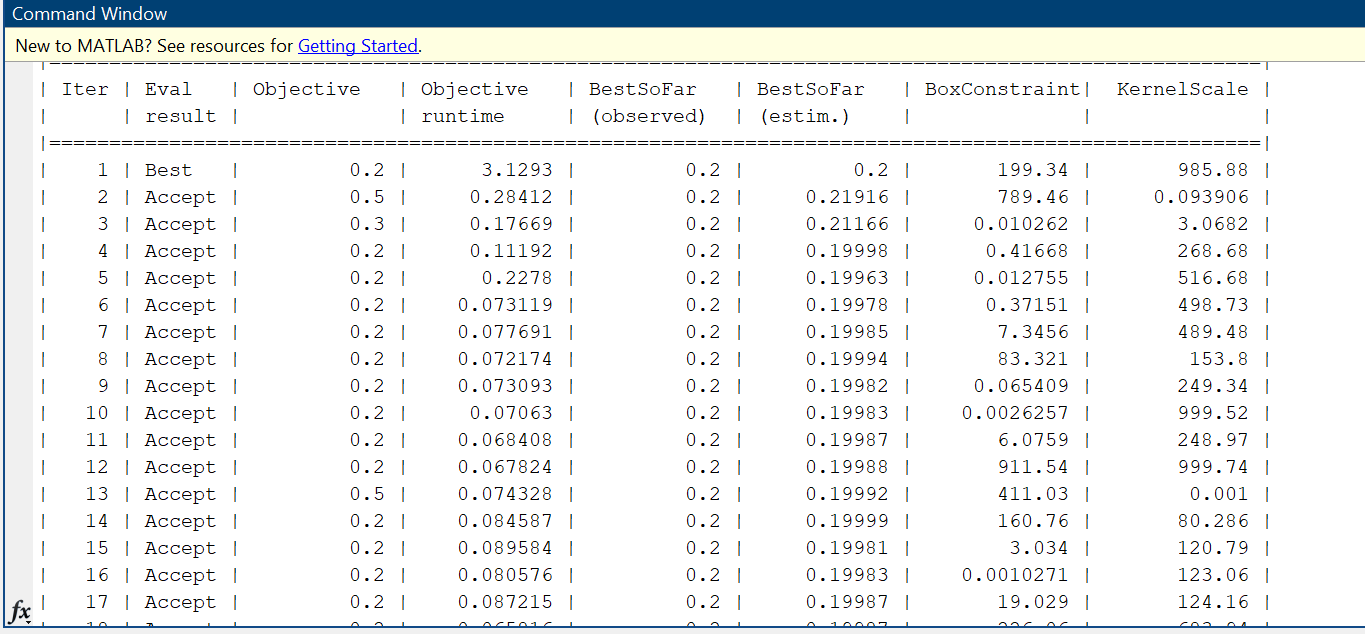
rng default

SVMModel = fitcsvm(X,y,'Standardize',true,'KernelFunction','RBF','OptimizeHyperparameters','auto',...

'HyperparameterOptimizationOptions',struct('AcquisitionFunctionName',...

'expected-improvement-plus'))

**Output :**



**Conclusion :** Thus, after performing this experiment on Support Vector Machine on English Alphabets I was able to classify 2 alphabets and observe the Objective function model & Min Objective vs No. of function evaluation curve with 10 observations and bias as 0.1700 on MATLAB.

**Experiment No : 9**

**Aim :**To apply PCA on n Dimension inbuilt dataset.

**Theory :**

Principal Component Analysis (PCA) is used to explain the variance-covariance structure of a set of variables through linear combinations. It is often used as a dimensionality-reduction technique. It is the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest. It is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. It does so by creating new uncorrelated variables that successively maximize variance.

**Dataset used :** Inbuilt hald dataset

**Code :**

clc;

clear all;

close all;

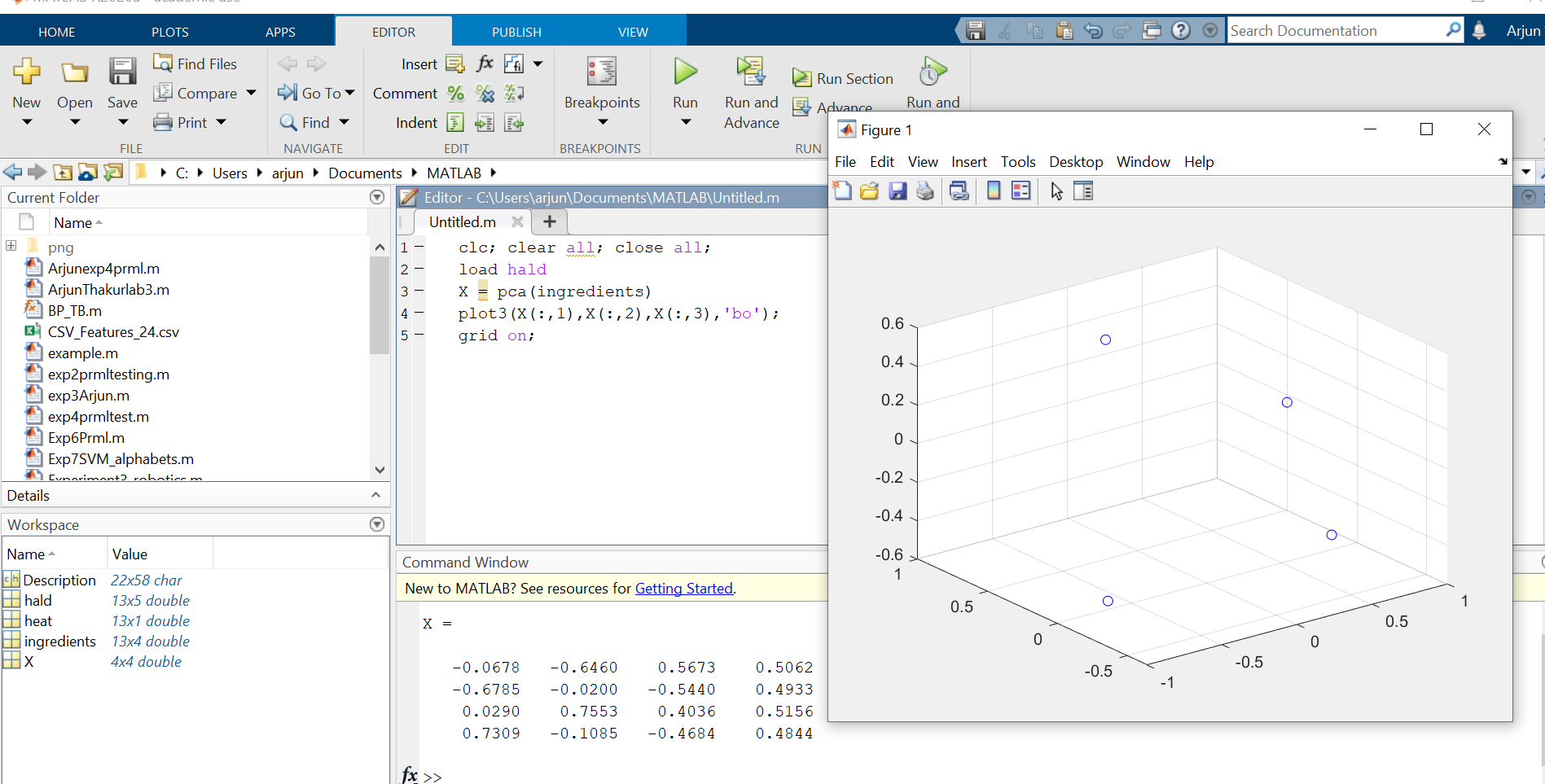
load hald

X = pca(ingredients)

plot3(X(:,1),X(:,2),X(:,3),'bo');

grid on;

**Output :**



**Conclusion** : Thus, after performing the experiment , I understood how Principle component analysis performs dimensionality reduction, how it  extracts the important information from a multivariate data table and to express this information as a set of few new variables called principal components on inbuilt dataset ‘hald” using inbuilt function “ pca (ingredients) “ & observing the X matrix in command window all successfully on MATLAB.

**Experiment No : 10**

**Aim:** To perform Principle Component Analysis (PCA) using Alphabet dataset with 24 features.

**Theory:** Principal Components Analysis (PCA) is an algorithm to transform the columns of a dataset into a new set of features called Principal Components. By doing this, a large chunk of the information across the full dataset is effectively compressed in fewer feature columns. This enables dimensionality reduction and ability to visualize the separation of classes or clusters if any.Principal Component Analysis is a dimensionality-reduction, it also transforms a large set of variables into a smaller one that still contains most of the information in the large set being completely different from the original dataset.

STEP BY STEP EXPLANATION OF PCA

### STEP 1: STANDARDIZATION

### STEP 2: COVARIANCE MATRIX COMPUTATION

### STEP 3: COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

**Dataset used :** Alphabet 24 feature dataset.

**Code:**

clc;

close all;

clear all;

x = csvread('Arjun \_exp9.csv');

[pcs,scrs,~,~,pexp] = pca(x);

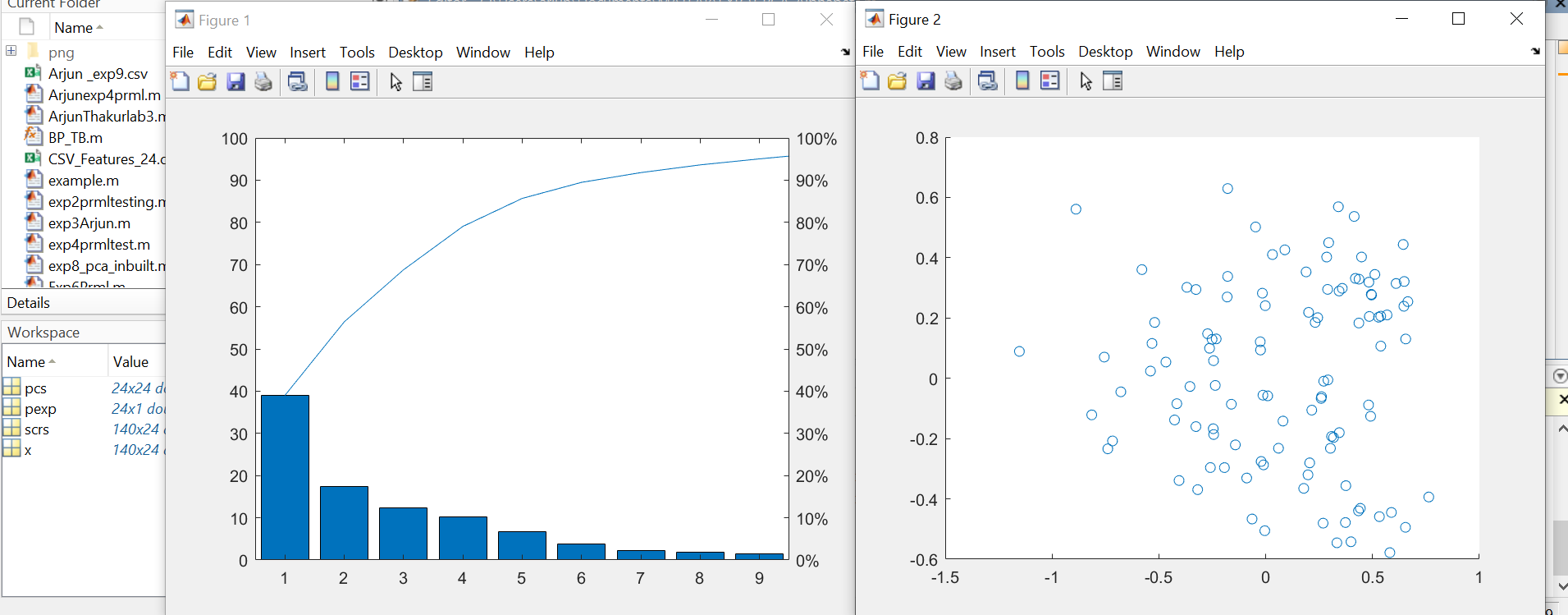
figure(1);

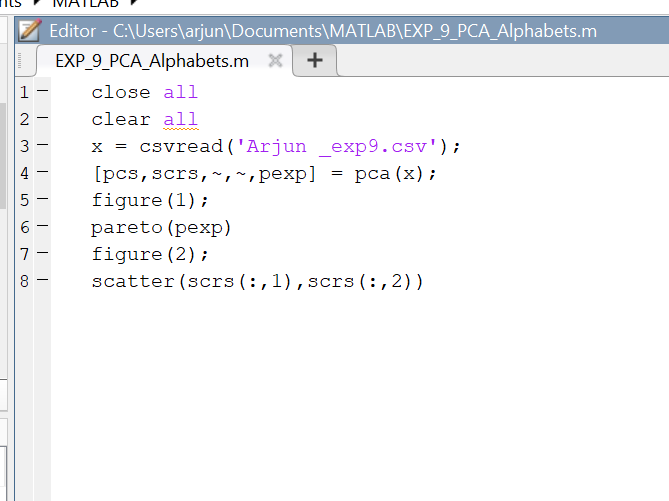
pareto(pexp)

figure(2);

scatter(scrs(:,1),scrs(:,2))

**Output :**





**Conclusion :** After performing this experiment of Principle Component Analysis (PCA) using Alphabet dataset, I observed the plot between Principle Component vs Percentage of experienced variances. Also the first principal component accounts for the **largest possible variance**in the data set. From the plot I observed that 95% and 1% relativity are present from the different groups with principle components. From fig 2 it is observed that maximum clusters lie between -0.5 to 0.5.

**Experiment No.11 (CNN)**

**Aim-** Pattern recognition application and Understanding of Convolutional Neural Network Algorithm.

**Pattern Recognition -** Pattern recognition is a process that looks at the available data and tries to see whether there are any regularities within it. There are two main parts:

1. Explorative part, where the algorithms are looking for patterns in general
2. Descriptive part, where the algorithms start to categorize the found patterns

Unlike with computer vision, the pattern recognition can be anything:

1. Texts or words
2. Images
3. Sentiments (emotions)
4. Sounds
5. Other elements and information

Pattern recognition is a process that takes in raw data and makes an action based on the category of the pattern. It optimally extracts patterns based on certain conditions and separates one class from another. Pattern recognition was often achieved using linear and quadratic discriminants, the k-nearest neighbour classifier or the Parzen density estimator, template matching and Neural Networks.

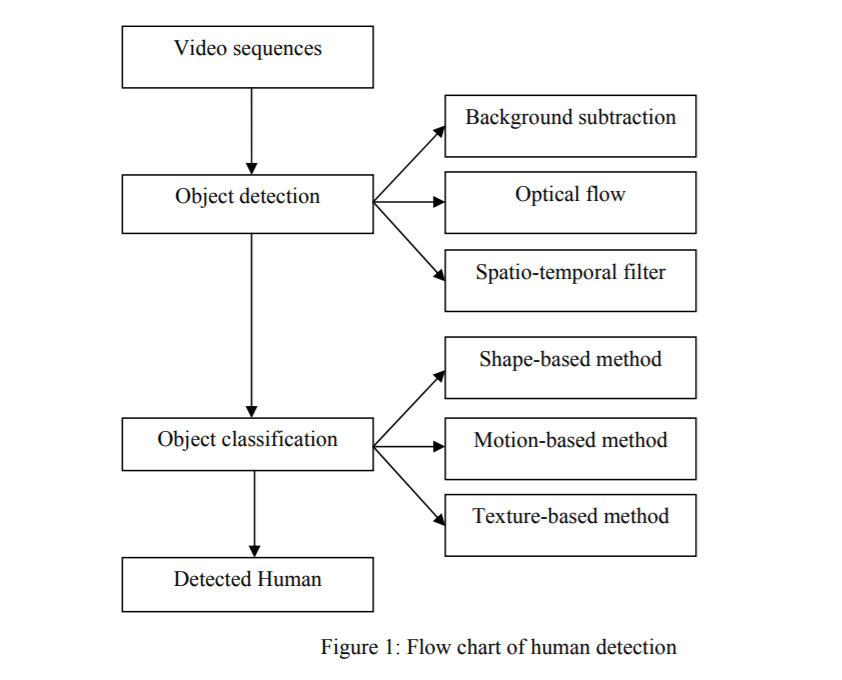
**Application of pattern recognition to be discussed-** Human detection in surveillance videos

**Introduction -** Detecting human beings in a video scene of a surveillance system is attracting more attention due to its wide range of applications in abnormal event detection, human gait characterization, person counting in a dense crowd, person identification, gender classification, fall detection for elderly people, etc. The scenes obtained from a surveillance video are usually with low resolution. Most of the scenes captured by a static camera are with minimal change of background. Objects in the outdoor surveillance are often detected in far field. Most existing digital video-surveillance systems rely on human observers for detecting specific activities in a real time video scene. But there are limitations in the human capability to monitor simultaneous events in surveillance displays.

Hence human motion analysis in automated video surveillance has become one of the most active and attractive research topics in the area of computer vision and pattern recognition. An intelligent system detects and captures motion information of moving targets for accurate object classification. The classified object is being tracked for high-level analysis.

**Techniques-** The detection process generally occurs in two steps: Object detection and Object classification. Object detection could be performed by background subtraction, optical flow and spatio-temporal filtering. The object classification methods could be divided into three categories: shape-based, motion-based and texture-based.

**Block Diagram –**



**Dataset used –**

1. KTH dataset is the largest available and most standard dataset widely used for benchmarking results for human action classification. The dataset contains six activities (boxing, hand waving, handclapping, running, jogging, and walking) performed by 25 subjects in four different scenarios.
2. Weizmann human action dataset contains a total of ten actions performed by nine people, to provide a total of 90 videos. Sample sequences are shown in Figure 5. The dataset contains videos with a static camera unlike KTH, where some of the videos had zooming and also the videos have simple background. As this dataset contains ten activities, which is more compared to six activities of KTH dataset

**Applications -** For an intelligent video surveillance system the detection of a human being is important for abnormal event detection, human gait characterization, people counting, person identification and tracking, pedestrian detection, gender classification, fall detection of elderly people, etc

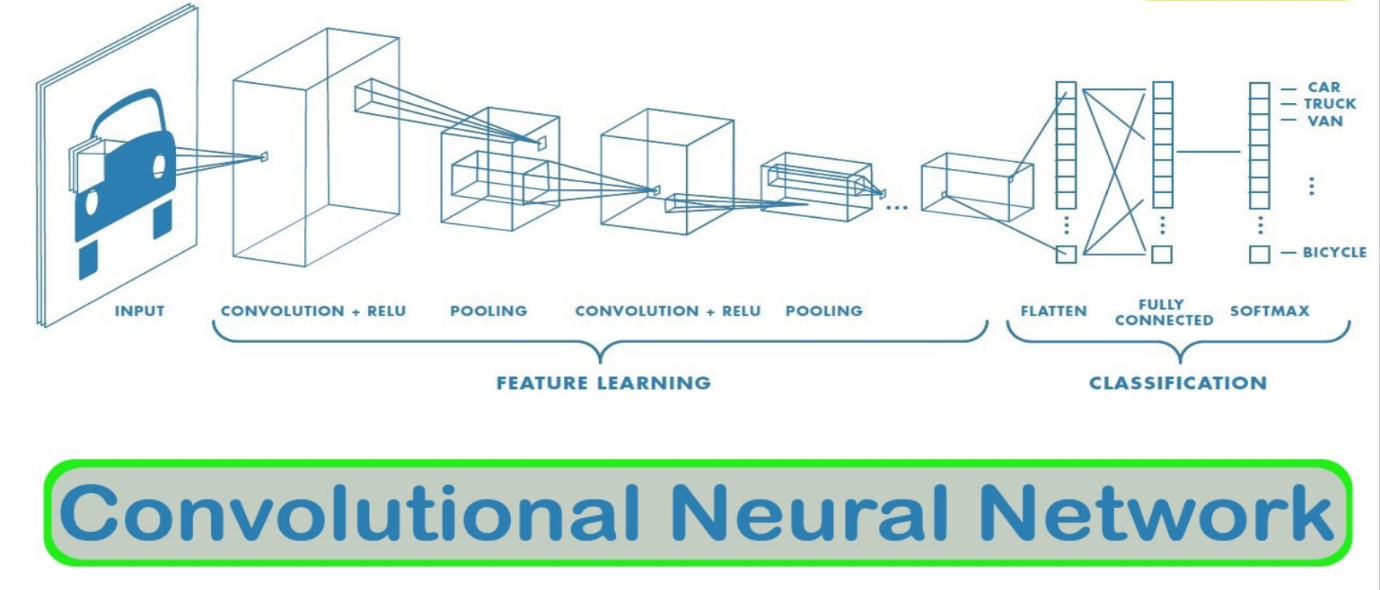
**Result -** A significant amount of work has been done with a view to detect human beings in a surveillance video. However the low-resolution images from the surveillance cameras always make this work challenging. Most of the object detection methods rely on known operation environment. The model adaptation speed based on observed scene statistics could be improved in future for faster adaptation of changed background and better persistency. But occlusion is a major problem for background segmentation technique. Optical flow and Spatio-Temporal filter techniques address this issue to some extent where the object of interest is occluded by a fixed object. But it is always difficult to detect an object in motion which is occluded by objects with similar shape and motion. One solution could be constructing 3D image for a 3D system by using volume information obtained from multiple cameras. From the machine vision perspective it is hard to distinguish an object as a human due to its large number of possible appearances. Moreover the motion of human is not always periodic.

**Conclusion -** Detecting human beings accurately in a surveillance video is one of the major topics of vision research due to its wide range of applications. It is challenging to process the image obtained from a surveillance video as it has low resolution. A review of the available detection techniques is presented. The detection process occurs in two steps: object detection and object classification. In this paper, all the available object detection techniques are categorised into background subtraction, optical flow and spatio-temporal filter method. The object classification techniques are categorised into shape-based, motion-based and texture-based method. The characteristics of the benchmark datasets are presented and major applications of human detection in surveillance video are reviewed.

**Introduction to CNN -** In [deep learning](https://en.wikipedia.org/wiki/Deep_learning), a convolutional neural network (CNN, or ConvNet) is a class of [deep neural networks](https://en.wikipedia.org/wiki/Deep_neural_network), most commonly applied to analyzing visual imagery. They have applications in [image and video recognition](https://en.wikipedia.org/wiki/Computer_vision), [recommender systems](https://en.wikipedia.org/wiki/Recommender_system), [image classification](https://en.wikipedia.org/wiki/Image_classification), [medical image analysis](https://en.wikipedia.org/wiki/Medical_image_computing), [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing), [brain-computer interfaces](https://en.wikipedia.org/wiki/Brain%E2%80%93computer_interface), and financial [time series](https://en.wikipedia.org/wiki/Time_series). CNNs are [regularized](https://en.wikipedia.org/wiki/Regularization_(mathematics)) versions of [multilayer perceptrons](https://en.wikipedia.org/wiki/Multilayer_perceptron). A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.

#### **Purpose of Convolutional Neural Network** - Mainly to process and analyse digital images, with some success cases involving processing voice and natural language.

**Structure (with an example of vehicle classification) –**



**The CNN is a combination of two basic building blocks -**

1. **The Convolution Block**— Consists of the Convolution Layer and the Pooling Layer. This layer forms the essential component of Feature-Extraction
2. **The Fully Connected Block**— Consists of a fully connected simple neural network architecture. This layer performs the task of Classification based on the input from the convolutional block.

**There are 4 major operations in CNN image detection/classification -**

1. Convolution
2. Activation map
3. Max pooling
4. Flattening
5. Fully connected layer

**Difference between ANN and CNN-** The major difference between a traditional Artificial Neural Network (ANN) and CNN is that only the last layer of a CNN is fully connected whereas in ANN, each neuron is connected to every other neurons.

#### **Advantage of using CNN -**

Little dependence on pre processing, decreasing the needs of human effort developing its functionalities.

**Conclusion :** After doing this task on CNN, I understood how Convolutional Neural Network works, it’s block diagram, working & I also performed CNN application based task on Human detection in surveillance videos regarding the same.