**EX-7**

1. **computeCentroids**

% DIMENSIONS:

% X = m x n

% centroids = K x n

%% %%%%%% WORKING: SOLUTION1 %%%%%%%%%

% for i = 1:K

% idx\_i = find(idx==i); %indexes of all the input which belongs to cluster j

% centroids(i,:)=(1/length(idx\_i))\*sum(X(idx\_i,:)); %calculating mean manually

% end

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

%% %%%%%% WORKING: SOLUTION 2 %%%%%%%%

for i = 1:K

idx\_i = find(idx==i); %indexes of all the input which belongs to cluster j

centroids(i,:) = mean(X(idx\_i,:)); % calculating mean using built-in function

end

1. **findClosesetCentroids**

% DIMENSIONS:

% centroids = K x no. of features = 3 x 2

for i = 1:size(X,1)

temp = zeros(K,1);

for j = 1:K

temp(j)=sqrt(sum((X(i,:)-centroids(j,:)).^2));

end

[~,idx(i)] = min(temp);

End

1. **kMeansInitCentroids**

% Randomly reorder the indices of examples

randidx = randperm(size(X, 1));

% Take the first K examples as centroids

centroids = X(randidx(1:K), :);

1. **Pca**

% DIMENSIONS :

% X = m x n

Sigma = (1/m)\*(X'\*X); % n x n

[U, S, V] = svd(Sigma);

1. **projectData**

% DIMENSIONS:

% X = m x n

% U = n x n

% U\_reduce = n x K

% K = scalar

U\_reduce = U(:,[1:K]); % n x K

Z = X \* U\_reduce; % m x k

1. **recoverData**

% DIMENSIONS:

% Z = m x K

% U = n x n

% U\_reduce = n x k

% K = scalar

% X\_rec = m x n

U\_reduce = U(:,1:K); % n x k

X\_rec = Z \* U\_reduce'; % m x n