

BÀI TẬP THỰC HÀNH 1

1. Load file ảnh đầu tiên lên màn hình

```
AllImage=loadMNISTImages('train-images.idx3-ubyte');
% 60.000 ảnh.
Image00001=reshape(AllImage(:,1),28,28);
imshow(Image00001);
```

2. Nạp dữ liệu train và dữ liệu test của bài toán nhận dạng chữ viết tay

```
function Reconition001_digits()
    fprintf('\n Load du lieu train');
    imgTrainAll=loadMNISTImages('train-images.idx3-ubyte');
    lblTrainAll=loadMNISTLabels('train-labels.idx1-ubyte');
    fprintf('\n Load du lieu test');
    imgTestAll=loadMNISTImages('t10k-images.idx3-ubyte');
    lblTestAll=loadMNISTLabels('t10k-labels.idx1-ubyte');
    fprintf('\n Ket thuc \n');
end
```

3. Số lượng ảnh train, số lượng ảnh test

```
function Reconition002_digits()
    fprintf('\n Load du lieu train');
    imgTrainAll=loadMNISTImages('train-images.idx3-ubyte');
    lblTrainAll=loadMNISTLabels('train-labels.idx1-ubyte');
    fprintf('\n Load du lieu test');
    imgTestAll=loadMNISTImages('t10k-images.idx3-ubyte');
    lblTestAll=loadMNISTLabels('t10k-labels.idx1-ubyte');

    nTrainImages=size(imgTrainAll,2);
    nTrainLabels=size(lblTrainAll,1);

    nTestImages=size(imgTestAll,2);
    nTestLabels=size(lblTestAll,1);

    nSizeofImage=size(imgTrainAll,1);

    fprintf('\n So luong anh train: [%d].',nTrainImages);
    fprintf('\n So luong nhan train: [%d].',nTrainLabels);
    fprintf('\n So luong anh test: [%d].',nTestImages);
    fprintf('\n So luong nhan test: [%d].',nTestLabels);
    fprintf('\n Kich thuoc cua mot anh: [%d].',nSizeofImage);

end
```

4. Hiện thị ảnh đầu, ảnh cuối

```
function Reconition003_digits()
```

```

fprintf('\n Load du lieu train');
imgTrainAll=loadMNISTImages('train-images.idx3-ubyte');
lblTrainAll=loadMNISTLabels('train-labels.idx1-ubyte');
fprintf('\n Load du lieu test');
imgTestAll=loadMNISTImages('t10k-images.idx3-ubyte');
lblTestAll=loadMNISTLabels('t10k-labels.idx1-ubyte');

nTrainImages=size(imgTrainAll,2);
figure;
img=imgTrainAll(:,1);
img2D=reshape(img,28,28);
strLabelImage=num2str(lblTrainAll(1));
imshow(img2D);
title(strLabelImage);

figure;
imgLast=imgTrainAll(:,nTrainImages);
img2DLast=reshape(imgLast,28,28);
strLabelImage=num2str(lblTrainAll(nTrainImages));
imshow(img2DLast);
title(strLabelImage);
end

```

5. Hiện thị ảnh ngẫu nhiên

```

function Reconition004_digits()
imgTrainAll=loadMNISTImages('train-images.idx3-ubyte');
lblTrainAll=loadMNISTLabels('train-labels.idx1-ubyte');

imgTestAll=loadMNISTImages('t10k-images.idx3-ubyte');
lblTestAll=loadMNISTLabels('t10k-labels.idx1-ubyte');

nTrainImages=size(imgTrainAll,2);
nNumber=randi([1 nTrainImages]);
figure;
img=imgTrainAll(:,nNumber);
img2D=reshape(img,28,28);
strLabelImage=num2str(lblTrainAll(nNumber));
strLabelImage=[strLabelImage, '(', num2str(nNumber), ')'];
imshow(img2D);
title(strLabelImage);

nTestImgs=size(imgTestAll,2);
nNumber=randi([1 nTestImgs]);
figure;
img=imgTestAll(:,nNumber);
img2D=reshape(img,28,28);
strLabelImage=num2str(lblTestAll(nNumber));
strLabelImage=[strLabelImage, '(', num2str(nNumber), ')'];
imshow(img2D);
title(strLabelImage);

end

```

6. Xây dựng model từ tập dữ liệu train bằng thuật toán kNN, nạp dữ liệu test. Kiểm tra kết quả tiên đoán.

```
function Reconition005_Digits_kNN()
    imgTrainAll=loadMNISTImages('train-images.idx3-ubyte');
    lblTrainAll=loadMNISTLabels('train-labels.idx1-ubyte');

    Mdl=fitcknn(imgTrainAll',lblTrainAll); %L?u ý ch? này.

    imgTestAll=loadMNISTImages('t10k-images.idx3-ubyte');
    lblTestAll=loadMNISTLabels('t10k-labels.idx1-ubyte');

    nTestImgs=size(imgTestAll,2);
    nNumber=randi([1 nTestImgs]);
    imgTest=imgTestAll(:,nNumber);
    lblPredictTest=predict(Mdl,imgTest'); %L?u ý ch? này.
    lblImageTest=lblTestAll(nNumber);
    figure;

    img2D=reshape(imgTest,28,28);
    imshow(img2D);

    strLabelImage='Ban dau';
    strLabelImage=[strLabelImage,num2str(lblTestAll(nNumber)),'.']; %L?u ý
ch? này.
    strLabelImage=[strLabelImage,'Du doan: '];
    strLabelImage=[strLabelImage,num2str(lblPredictTest),'.'];

    if(lblPredictTest==lblImageTest)
        strLabelImage=[strLabelImage,'Ket qua dung '];
    else
        strLabelImage=[strLabelImage,'Ket qua sai '];
    end
end
```

BÀI TẬP THỰC HÀNH 2

1. nNumber=randi([1 200]);
2. A(3,5)
3. A= zeros[100 200]
4. d = size (x)
 - [m,n] = size (x)
 - m = size (x,1) % số dòng
 - n = size (x,2) % số cột
5. A(:,10)
6. A(10,:)
7. A1=reshape (A,28,28);

BÀI TẬP THỰC HÀNH 3

Q1. Hãy viết function hiển thị ảnh có thứ tự là n (n là tham số) cùng label tương ứng trong tập huấn luyện (train) của tập dữ liệu MNIST. Paste code vào bài thực hành và lập bảng cho biết kết quả khi chạy với n=1, 500, 5000, 10000, 59000.

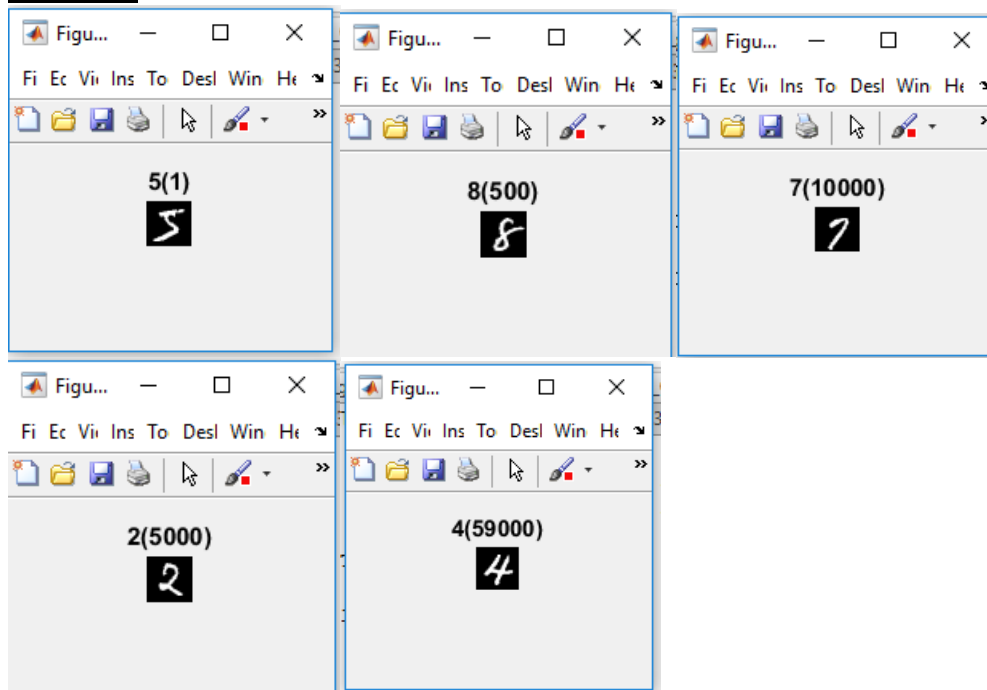
```
function displayimage(n)
    imgTrainAll=loadMNISTImages('train-images.idx3-ubyte');
    lblTrainAll=loadMNISTLabels('train-labels.idx1-ubyte');

    Mdl=fitcknn(imgTrainAll',lblTrainAll);

    nTrainImages=size(imgTrainAll,2);

    img=imgTrainAll(:,n);
    img2D=reshape(img,28,28);
    strLabelImage=num2str(lblTrainAll(n));
    strLabelImage=[strLabelImage, '(', num2str(n), ')'];
    imshow(img2D);
    title(strLabelImage);
end
```

Kết quả:



Q2. Hãy viết function hiển thị ảnh có thứ tự là n (n là tham số) cùng label tương ứng trong tập test của tập dữ liệu MNIST. Paste code vào bài thực hành và lập bảng cho biết kết quả khi chạy với n=1, 500, 5000, 9000.

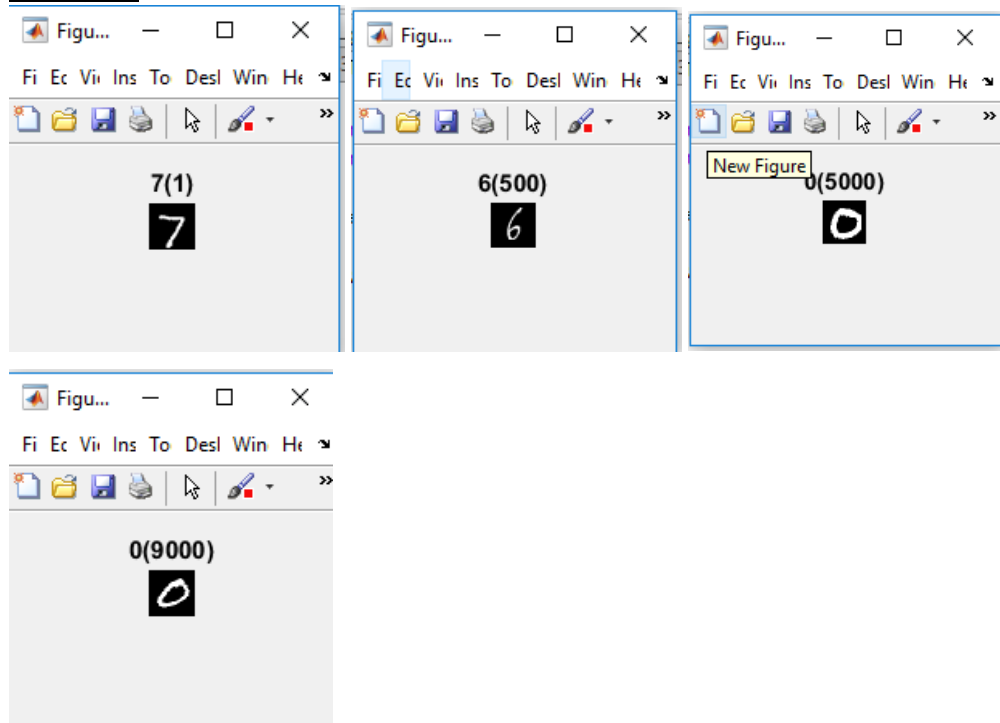
```
function displayimageTest(n)
    imgTestAll=loadMNISTImages('t10k-images.idx3-ubyte');
    lblTestAll=loadMNISTLabels('t10k-labels.idx1-ubyte');

    Mdl=fitcknn(imgTestAll',lblTestAll);

    nTestImgs=size(imgTestAll,2);
    nNumber=n;
    figure;
    img=imgTestAll(:,nNumber);
    img2D=reshape(img,28,28);
    strLabelImage=num2str(lblTestAll(nNumber));
    strLabelImage=[strLabelImage,'(',num2str(nNumber),')'];
    imshow(img2D);
    title(strLabelImage);
end
```

end

Kết quả:



Q3. Hãy viết function thống kê số lượng các ảnh tương ứng với các label trong tập huấn luyện (train) của tập dữ liệu MNIST. Paste code vào bài thực hành và lập bảng kết quả khi chạy (nên xuất dưới dạng csv để tiện import thành bảng).

```
function StatictisTrain(n)

    lblTrainAll=loadMNISTLabels('train-labels.idx1-ubyte');
    nTrainLabels = size(lblTrainAll);

    nResult = zeros(10,2);
    for i = 1 : nTrainLabels
        k = lblTrainAll(i);
        nResult(k+1,1) = nResult(k+1,1) + 1;
    end

    fprintf('Thong ke:\n');
    for i = 1:10
        x = i - 1;
        nResult(i,1) = x;
        nResult(i,2) = sum(lblTrainAll == x);
        fprintf('So anh co nhan %d: %d\n', i-1, nResult(i,2));
    end

    disp(nResult); % Trinh bay noi dung cua bien ra man hinh.

    % luu file csv
    strFileName = ['E:\Q3', '.csv'];
    csvwrite(strFileName, nResult);

end
```

Kết quả:

Thong ke:	
So anh co nhan 0: 5923	<u>0</u> <u>5923</u>
So anh co nhan 1: 6742	<u>1</u> <u>6742</u>
So anh co nhan 2: 5958	<u>2</u> <u>5958</u>
So anh co nhan 3: 6131	<u>3</u> <u>6131</u>
So anh co nhan 4: 5842	<u>4</u> <u>5842</u>
So anh co nhan 5: 5421	<u>5</u> <u>5421</u>
So anh co nhan 6: 5918	<u>6</u> <u>5918</u>
So anh co nhan 7: 6265	<u>7</u> <u>6265</u>
So anh co nhan 8: 5851	<u>8</u> <u>5851</u>
So anh co nhan 9: 5949	<u>9</u> <u>5949</u>

Q4. Hãy viết function thống kê số lượng các ảnh tương ứng với các label trong tập test của tập dữ liệu MNIST. Paste code vào bài thực hành và lập bảng kết quả khi chạy (nên xuất dưới dạng csv để tiện import thành bảng).

```
imgTestAll=loadMNISTImages('t10k-images.idx3-ubyte');
lblTestAll=loadMNISTLabels('t10k-labels.idx1-ubyte');
nTestImgs=size(imgTestAll,2);

nTestLabels = size(lblTestAll);

nResult = zeros(10);

for i = 1 : nTestLabels
    k = lblTestAll(i);
    nResult(k+1) = nResult(k+1) + 1;
end

fprintf('Thong ke:\n');

for i = 1:10
    fprintf('So anh co nhan %d: %d\n', i-1, nResult(i));
end

end
```

Kết quả:

Thong ke:

So anh co nhan 0: 980

So anh co nhan 1: 1135

So anh co nhan 2: 1032

So anh co nhan 3: 1010

So anh co nhan 4: 982

So anh co nhan 5: 892

So anh co nhan 6: 958

So anh co nhan 7: 1028

So anh co nhan 8: 974

So anh co nhan 9: 1009

Q5. Hãy viết function trả về kết quả nhận dạng của ảnh trong tập test có thứ tự là n (n là tham số, nằm trong đoạn [1, 10000]). Paste code vào bài thực hành và lập bảng kết quả khi chạy với n = 5, 500, 900.

```
function TH3_Q5(n)
    imgTestAll = loadMNISTImages('t10k-images.idx3-ubyte');
    nTestImgs = size(imgTestAll);

    imgTrainAll=loadMNISTImages('train-images.idx3-ubyte');
    lblTrainAll=loadMNISTLabels('train-labels.idx1-ubyte');
    Mdl = fitcknn(imgTrainAll', lblTrainAll);
    imgTest = imgTestAll(:,n);
    lblPredictTest = predict(Mdl, imgTest');

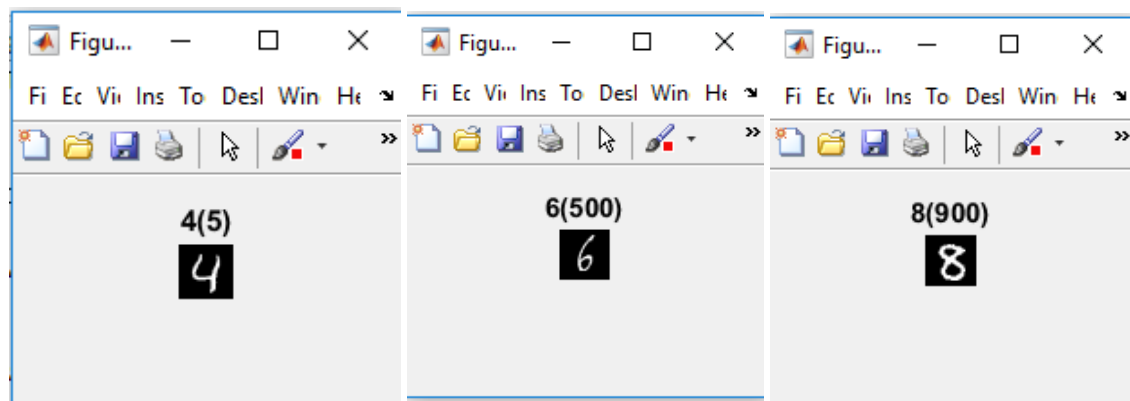
    figure;
    img2D=reshape(imgTest,28,28);
    imshow(img2D);

    strLabelImage=num2str(lblPredictTest);
    strLabelImage=[strLabelImage, '(', num2str(n), ')'];
    title(strLabelImage);

    fprintf('Anh test thu: %d\nDuoc du doan la chu so: %d.\n',n,lblPredictTest);

end
```

Kết quả:



Q6. Hãy viết function với tham số đầu vào n là thứ tự của ảnh trong tập test - sau đó hiển thị ảnh tương ứng - rồi hiển thị kết quả nhận dạng - rồi cho biết kết quả nhận dạng là đúng hay sai khi so khớp với label của tập test.

```
function TH3_Q6(n)
    strData='train-images.idx3-ubyte';
    strDataLabel='train-labels.idx1-ubyte';
    [imgDataTrain, lblDataTrain]=loadData(strData, strDataLabel);

    strData='t10k-images.idx3-ubyte';
    strDataLabel='t10k-labels.idx1-ubyte';
    [imgDataTest, lblDataTest]=loadData(strData, strDataLabel);

    Mdl=fitcknn(imgDataTrain', lblDataTrain);

    nTestImgs=size(imgDataTest, 2); % 10.000 anh test.
    nNumber=n;
    imgTest=imgDataTest(:, nNumber);

    lblPredictTest=predict(Mdl, imgTest'); %Ma tran chuyen vi, chuyen anh
    tren 1 dong.
    lblImageTest = lblDataTest(n);
    figure;

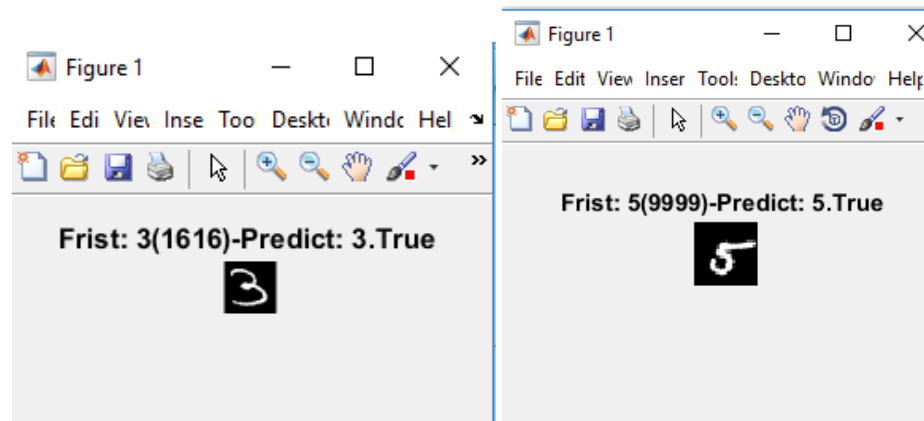
    img2D=reshape(imgTest, 28, 28);
    imshow(img2D);

    strLabelImage='Frist ';
    strLabelImage=[strLabelImage, num2str(lblImageTest)]; %Luu y
    strLabelImage=[strLabelImage, '(', num2str(n), ')']; % Thu tu
    strLabelImage=[strLabelImage, 'Predict: '];
    strLabelImage=[strLabelImage, num2str(lblPredictTest), '.'];

    if(lblPredictTest==lblImageTest)
        strLabelImage=[strLabelImage, 'True '];
    else
        strLabelImage=[strLabelImage, 'False '];
    end

    title(strLabelImage);

end
```

Kết quả:

Q6*. Hãy viết thêm phần giao diện cho bài trên - tham khảo code ở đây:

<https://bitbucket.org/intelligenceagent/cudacnn-public/wiki/Home>

```
function varargout = TH3_Q6x(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',    mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @TH3_Q6x_OpeningFcn, ...
    'gui_OutputFcn', @TH3_Q6x_OutputFcn, ...
    'gui_LayoutFcn', [] , ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

```

end

function TH3_Q6x_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

guidata(hObject, handles);

function varargout = TH3_Q6x_OutputFcn(hObject, eventdata, handles)

varargout{ 1 } = handles.output;

function nInput_Callback(hObject, eventdata, handles)

function nInput_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function pushbutton1_Callback(hObject, eventdata, handles)

n = str2double(get(handles.nInput,'string'));
imgTrainAll = loadMNISTImages('train-images.idx3-ubyte');
lblTrainAll = loadMNISTLabels('train-labels.idx1-ubyte');
```

```

Mdl = fitcknn(imgTrainAll' , lblTrainAll);

imgTestAll = loadMNISTImages('t10k-images.idx3-ubyte');
lblTestAll = loadMNISTLabels('t10k-labels.idx1-ubyte');
imgTest = imgTestAll(:, n);
lblPredictTest = predict(Mdl, imgTest');
lblImageTest = lblTestAll(n);

img2D = reshape(imgTest, 28, 28);
imshow(img2D,'Goc',handles.showimage);
strLabelImage = 'Ban dau ';
strLabelImage = [strLabelImage, num2str(lblImageTest), '.'];
strLabelImage = [strLabelImage, ' Du doan: '];
strLabelImage = [strLabelImage, num2str(lblPredictTest), '.'];

if(lblPredictTest == lblImageTest)
    strLabelImage = [strLabelImage, ' Nhan dang dung.'];
else
    strLabelImage = [strLabelImage, ' Nhan dang sai.'];
end
set(handles.result,'string',strLabelImage);

```

Q7. Hãy viết function đếm số lượng các ảnh có label là n (n là tham số) bị nhận dạng sai theo thuật toán knn. Paste code của function đã chạy được vào bài thực hành và lập bảng kết quả khi chạy với n= 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

```
function TH3_Q7(n)
    strData='train-images.idx3-ubyte';
    strDataLabel='train-labels.idx1-ubyte';
    [imgDataTrain, lblDataTrain]=loadData(strData, strDataLabel);
    strData='t10k-images.idx3-ubyte';
    strDataLabel='t10k-labels.idx1-ubyte';
    [imgDataTest, lblDataTest]=loadData(strData, strDataLabel);
    Mdl=fitcknn(imgDataTrain', lblDataTrain);
    %ClassNames: [0 1 2 3 4 5 6 7 8 9].
    x=size(imgDataTest,2);
    % 10.000 anh test, so luong anh Test.
    y = []; % Mang chua so luong anh Test.
    for i = 1:x
        if (lblDataTest(i) == n)
            y = [y, imgDataTest(:, i)];
        end
    end
    countFalse = 0;
    for i = 1:size(y,2)
        imgTest = y(:, i);
        lblPredictTest = predict(Mdl, imgTest');
        if (lblPredictTest ~= n)
            countFail = countFalse + 1;
        end
    end
    fprintf('So anh co nhan %d bi nhan dang sai la %d\n', n,
countFalse);
end
```

Kết quả:

Label	Số lượng ảnh sai
0	7
1	6
2	40
3	40
4	38
5	32
6	14
7	36
8	54
9	42
SUM	309

Q7*. Tương tự bài tập Q7 - nhưng lập bảng confusion matrix.

```
function TH3_Q7x()
    strData='train-images.idx3-ubyte';
    strDataLabel='train-labels.idx1-ubyte';
    [imgDataTrain, lblDataTrain]=loadData(strData, strDataLabel);

    strData='t10k-images.idx3-ubyte';
    strDataLabel='t10k-labels.idx1-ubyte';
    [imgDataTest, lblDataTest]=loadData(strData, strDataLabel);

    Mdl=fitcknn(imgDataTrain', lblDataTrain);
    %ClassNames: [0 1 2 3 4 5 6 7 8 9].

    x=size(imgDataTest,2); % 10.000 anh test, so luong anh Test.

    A=zeros(10,10);

    for i=1:x
        lbla = lblDataTest(i);
        imga = imgDataTest(:, i);
        lblPredicta = predict(Mdl, imga');
        A(lbla + 1, lblPredicta + 1) = A(lbla + 1, lblPredicta + 1) + 1;
    end
    disp(A); % Trình bày nội dung của biến A ra màn hình.

    strFileName = ['E:\Q7x', '.csv'];
    csvwrite(strFileName, A);
end
```

Kết quả:

	0	1	2	3	4	5	6	7	8	9	TRUE	FALSE
0	973	1	1	0	0	1	3	1	0	0	973	7
1	0	112	3	0	1	1	1	0	0	0	1129	6
2	7	6	992	5	1	0	2	16	3	0	992	40
3	0	1	2	970	1	19	0	7	7	3	970	40
4	0	7	0	0	944	0	3	5	1	22	944	38
5	1	1	0	12	2	860	5	1	6	4	860	32
6	4	2	0	0	3	5	944	0	0	0	944	14
7	0	14	6	2	4	0	0	992	0	10	992	36
8	6	1	3	14	5	13	3	4	920	5	920	54
9	2	5	1	6	10	5	1	11	1	967	967	42
											9691	309
Tổng số anh Test											96.91 %	3.09 %

Q8.** (Tương đương 50% đồ án môn học nếu kết hợp với Q7*) - Hãy viết function tính độ chính xác của thuật toán knn với các tham số khác nhau của hàm knn (ví dụ số lượng nearest neighbors, độ đo distance). Paste code vào bài thực hành và lập bảng kết quả.

```
%https://www.mathworks.com/help/stats/classificationknn-class.html
%% Mdl = fitcknn(X,Y,'NumNeighbors',5,'Standardize',1)
% Y – Class labels
% numeric vector | categorical vector | logical vector | character
array | cell array of character vectors
% X – Predictor data
% numeric matrix
% X = meas;
% Y = species;
% Mdl = fitcknn(X,Y,'NumNeighbors',4);
%% Methods
% compareHoldout    Compare accuracies of two models using new data
% crossval          Cross-validated k-nearest neighbor classifier
% edge              Edge of k-nearest neighbor classifier
% loss              Loss of k-nearest neighbor classifier
% margin            Margin of k-nearest neighbor classifier
% predict           Predict labels using k-nearest neighbor classification
model
% resubEdge         Edge of k-nearest neighbor classifier by resubstitution
% resubLoss         Loss of k-nearest neighbor classifier by resubstitution
% resubMargin       Margin of k-nearest neighbor classifier by
resubstitution
% resubPredict      Predict resubstitution response of k-nearest neighbor
classifier
%% Value           Description
% 'cityblock'       City block distance.
% 'chebychev'       Chebychev distance (maximum coordinate difference).
% 'correlation'     One minus the sample linear correlation between
observations (treated as sequences of values).
% 'cosine'          One minus the cosine of the included angle between
observations (treated as vectors).
% 'euclidean'       Euclidean distance.
% 'hamming'         Hamming distance, percentage of coordinates that differ.
% 'jaccard'         One minus the Jaccard coefficient, the percentage of
nonzero coordinates that differ.
% 'mahalanobis'     Mahalanobis distance, computed using a positive
definite covariance matrix C. The default value of C is the sample
covariance matrix of X, as computed by nancov(X). To specify a
different value for C, set the DistParameter property of mdl using dot
notation.
% 'minkowski'       Minkowski distance. The default exponent is 2. To
specify a different exponent, set the DistParameter property of mdl
using dot notation.
% 'seuclidean'      Standardized Euclidean distance. Each coordinate
difference between X and a query point is scaled, meaning divided by a
scale value S. The default value of S is the standard deviation
computed from X, S = nanstd(X). To specify another value for S, set
the DistParameter property of mdl using dot notation.
```

```

% 'spearman'    One minus the sample Spearman's rank correlation
between observations (treated as sequences of values).
%
%% NumNeighbors=n; Distance=d;

function TH3_Q8(n, d)

    strData='train-images.idx3-ubyte';
    strDataLabel='train-labels.idx1-ubyte';
    [imgDataTrain, lblDataTrain]=loadData(strData, strDataLabel);

    strData='t10k-images.idx3-ubyte';
    strDataLabel='t10k-labels.idx1-ubyte';
    [imgDataTest, lblDataTest]=loadData(strData, strDataLabel);

    Mdl=fitcknn(imgDataTrain', lblDataTrain, 'NumNeighbors',
n, 'Distance', d);
    %ClassNames: [0 1 2 3 4 5 6 7 8 9].

    x=size(imgDataTest,2); % 10.000 anh test, so luong anh Test.

    A=zeros(10,10);

    for i=1:x
        lbla = lblDataTest(i);
        imga = imgDataTest(:, i);
        lblPredicta = predict(Mdl, imga');
        A(lbla + 1, lblPredicta + 1) = A(lbla + 1, lblPredicta + 1) +
1;
    end

    countTrue=0;
    countTrue= sum(lblPredicta==lblDataTest);
    rateTrue= countTrue/x*100;

end

```

Kết quả:

n	'euclidean'	'cosine'
1	96.91	97.23
3	97.06	97.37