

7.

A. ~

- Source Code

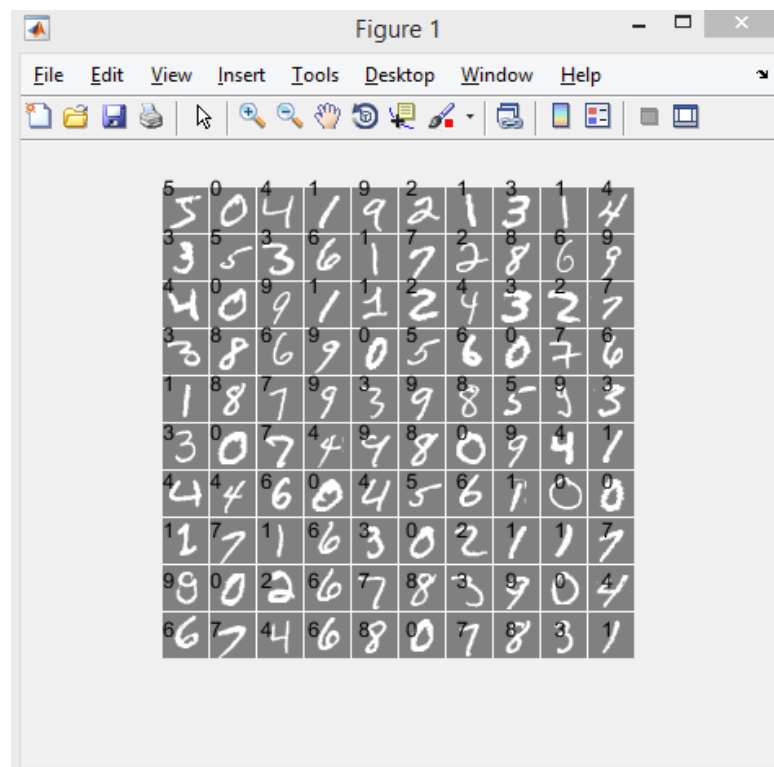
```
%menjalankan fungsi loadmnist
%5000 data image diextract ke vector
[X y] = loadmnist(5000);

%plot digitalhandwiring sebanyak 100 image
VariableTest = X(1:100,:);
[rows col] = size(VariableTest);
visual(VariableTest);
kolom = ceil(sqrt(size(VariableTest,1)));

yinvers = y(1:rows,:)';
baris = ceil(size(yinvers,2)/kolom);
newmatriks = reshape(yinvers,[kolom baris]);

%membuat text pada plot untuk memverifikasi data
%apakah sudah sesuai dengan label
for i=0:baris-1
    for j = 0 :kolom-1
        temp= mat2str(newmatriks(i+1,j+1))
        text(30*i,30*j,temp);
    end
end
```

- Verifikasi Data dengan label



- Screenshoot codingan

```
1      %menjalankan fungsi loadmnist
2      %5000 data image diextract ke vector
3 -    [X y] = loadmnist(5000);
4
5      %plot digitalhandwiring sebanyak 100 image
6 -    VariableTest = X(1:100,:);
7 -    [rows col] = size(VariableTest);
8 -    visual(VariableTest);
9 -    kolom = ceil(sqrt(size(VariableTest,1)));
10
11 -    yinvers = y(1:rows,:)';
12 -    baris = ceil(size(yinvers,2)/kolom);
13 -    newmatriks = reshape(yinvers,[kolom baris]);
14
15      %membuat text pada plot untuk memverifikasi data
16      %apakah sudah sesuai dengan label
17 -    for i=0:baris-1
18 -        for j = 0 :kolom-1
19 -            temp= mat2str(newmatriks(i+1,j+1))
20 -            text(30*i,30*j,temp);
21 -        end
22 -    end
```

- Analisis Program

Script diatas meload data handwriting sebanyak 5000 data kedalam matrix X dan matrix Y. matrix X merupakan data gambar yang masing-masing data barisnya terdiri dari 784 kolom. Adapun Y adalah label dari setiap baris dari matrix X. Untuk memvisualkan setiap data, menjalankan syntax visual(VariableTest) Kemudian untuk verifikasi, dilakukan dengan menambah label text pada setiap digit.

B. ~

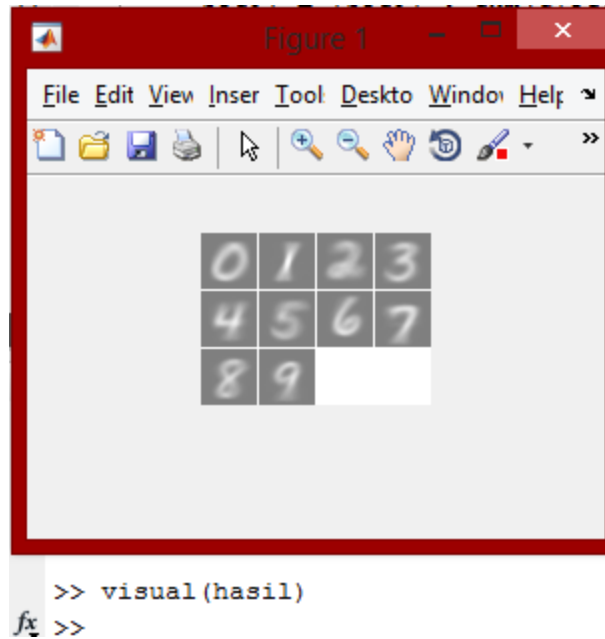
- Source Code & Penjelasan

```
function prototype = prototype( dataTraining , dataLabel)
%UNTITLED6 Summary of this function goes here
% Detailed explanation goes here
dataTraining = X;
dataLabel = y;
dataTrain_label = [dataTraining(1:2500,:) dataLabel(1:2500)];
dataTestiing_label = [dataTraining(2501:5000,:)
dataLabel(2501:5000)];
```

```
%membuat prototype
class0 = [];
class1 = [];
class2 = [];
class3 = [];
class4 = [];
class5 = [];
class6 = [];
class7 = [];
class8 = [];
class9 = [];
% membuat kelas dari dataTraining dengan label
%mengelompokkan data & menyesuaikan class dengan label
for j = 1:length(dataTrain_label)
    switch dataTrain_label(j,end)
        case 0
            class0 = [class0; dataTrain_label(j,:)];
        case 1
            class1 = [class1; dataTrain_label(j,:)];
        case 2
            class2 = [class2; dataTrain_label(j,:)];
        case 3
            class3 = [class3 ; dataTrain_label(j,:)];
        case 4
            class4 = [class4 ; dataTrain_label(j,:)];
        case 5
            class5 = [class5; dataTrain_label(j,:)];
        case 6
            class6 = [class6; dataTrain_label(j,:)];
        case 7
            class7 = [class7; dataTrain_label(j,:)];
        case 8
            class8 = [class8; dataTrain_label(j,:)];
        case 9
            class9 = [class9; dataTrain_label(j,:)];
    end
end
%hitung rata rata dari setiap class
hasil = sum(class0) / length(class0);
hasil = [hasil ; sum(class1) / length(class1)];
hasil = [hasil ; sum(class2) / length(class2)];
hasil = [hasil ; sum(class3) / length(class3)];
hasil = [hasil ; sum(class4) / length(class4)];
hasil = [hasil ; sum(class5) / length(class5)];
hasil = [hasil ; sum(class6) / length(class6)];
hasil = [hasil ; sum(class7) / length(class7)];
hasil = [hasil ; sum(class8) / length(class8)];
hasil = [hasil ; sum(class9) / length(class9)];

%hapus kolom terakhir (label);
hasil(:,end) = [];
end
```

- ScreenShoot hasil



- ScreenShoot codingan

```
1 function hasil = prototype( dataTraining , dataLabel)
2 %UNTITLED6 Summary of this function goes here
3 % Detailed explanation goes here
4 %dataTraining = X;
5 %dataLabel = y;
6 dataTrain_label = [dataTraining(1:2500,:) dataLabel(1:2500)];
7 dataTesting_label = [dataTraining(2501:5000,:) dataLabel(2501:5000)];
8
9
10 %membuat prototype
11 class0 = []; class1 = []; class2 = []; class3 = []; class4 = [];
12 class5 = []; class6 = []; class7 = []; class8 = []; class9 = [];
13 % membuat kelas dari dataTraining dengan label
14 %mengelompokkan data & menyesuaikan class dengan label
15 for j = 1:length(dataTrain_label)
16     switch dataTrain_label(j,end)
17         case 0
18             class0 = [class0; dataTrain_label(j,:)];
19         case 1
20             class1 = [class1; dataTrain_label(j,:)];
21         case 2
22             class2 = [class2; dataTrain_label(j,:)];
23         case 3
24             class3 = [class3 ; dataTrain_label(j,:)];
25         case 4
26             class4 = [class4 ; dataTrain_label(j,:)];
```

```
27 -         case 5
28 -             class5 = [class5; dataTrain_label(j,:)];
29 -         case 6
30 -             class6 = [class6; dataTrain_label(j,:)];
31 -         case 7
32 -             class7 = [class7; dataTrain_label(j,:)];
33 -         case 8
34 -             class8 = [class8; dataTrain_label(j,:)];
35 -         case 9
36 -             class9 = [class9; dataTrain_label(j,:)];
37 -     end
38 - end
39 - %hitung rata rata dari setiap class
40 - hasil = sum(class0) / length(class0);
41 - hasil = [hasil ; sum(class1) / length(class1)];
42 - hasil = [hasil ; sum(class2) / length(class2)];
43 - hasil = [hasil ; sum(class3) / length(class3)];
44 - hasil = [hasil ; sum(class4) / length(class4)];
45 - hasil = [hasil ; sum(class5) / length(class5)];
46 - hasil = [hasil ; sum(class6) / length(class6)];
47 - hasil = [hasil ; sum(class7) / length(class7)];
48 - hasil = [hasil ; sum(class8) / length(class8)];
49 - hasil = [hasil ; sum(class9) / length(class9)];
50 -
51 - %hapus kolom terakhir (label);
52 - hasil(:,end) = [];
53 - end
```

- Analisis Program

Fungsi diatas adalah fungsi untuk membuat suatu prototype dari sekumpulan data. Prototype diperoleh dari mengelompokkan semua data kedalam classnya. Kemudian menjumlahkan semua kolom dan merata-ratakan nilai pada kolom tersebut, maka diperoleh prototype setiap kelas.

C. ~

- Source Code & Penjelasan

```
function hasil = prototypebasedclassifier( dataTesting_label ,
dataPrototype )
%dataTrain = dataTrain_label; %sample
%dataPrototype = hasil;
cfsMat = zeros(10);
for i = 1:length(dataTesting_label);
    vectorEuclidean = [];

    %mencari nilai jarak euclidean dengan membandingkan setiap
    data oleh
```

```
%data prototype
for j = 1:10
    jarakEuclidean = norm(dataTestiing_label(i,1:end-1) -
dataPrototype(j,:));
    vectorEuclidean = [vectorEuclidean ; jarakEuclidean];
end

[M idx] = min(vectorEuclidean);
posisi = idx;
%membuat confusion matrix
cfsMat(dataTestiing_label(i,end)+1 , posisi) =
cfsMat(dataTestiing_label(i,end)+1 , posisi) + 1;
end

hasil = cfsMat;
end
```

- Hasil prototype classifier

```
>> hasil = prototypebasedclassifier(dataTestiing_label, dataPrototype)

hasil =

    236     0     0     1     0     0     8     0     0     0
     0    270     1     1     0     0     4     0     9     1
    10     4    193     7     5     0    10     1    11     0
    13     3     5    213     2     1     1     5     7     3
     1     1     0     0    228     0     8     3     4    10
    83     7     5     42    10    14    10    12    17     7
     6     0    13     0     4     0    221     0     3     0
    12     7     2     2    11     0     0    236     2     3
    18     2     7    42     6     0     6    10    146     3
    12     3     2     2    84     0     1    38     7    102
```

- Screenshoot codingan

```
1 function hasil = prototypebasedclassifier( dataTestiing_label , dataPrototype )
2 %dataTrain = dataTestiing_label; %sample
3 %dataPrototype = hasil;
4 cfsMat = zeros(10);
5 for i = 1:length(dataTestiing_label);
6     vectorEuclidean = [];
7
8     %mencari nilai jarak euclidean dengan membandingkan setiap data oleh
9     %data prototype
10    for j = 1:10
11        jarakEuclidean = norm(dataTestiing_label(i,1:end-1) - dataPrototype(j,:));
12        vectorEuclidean = [vectorEuclidean ; jarakEuclidean];
13    end
14
15    [M idx] = min(vectorEuclidean);
16    posisi = idx;
17    %membuat confusion matrix
18    cfsMat(dataTestiing_label(i,end)+1 , posisi) = cfsMat(dataTestiing_label(i,
19 end
20
21     hasil = cfsMat;
22 end
23
24
```

- Analisis Program

Fungsi diatas untuk memperoleh hasil confusion matrix yang berisi banyaknya data yang sesuai dengan classnya ataupun yang tidak sesuai dengan classnya. Untuk memperoleh confusion matrix dari setiap class, terlebih dahulu menghitung Euclidean antara seluruh data testing dengan setiap data prototype. Kemudian nilai minimal pada vector Euclidean menjadi nilai class pada data tersebut, dan di masukkan ke dalam confusion matrix sesuai kolom dan barisnya.

D. ~

- Source Code

```
function hasil = nearestneighbor( dataTestiing_label ,
dataTrain_label )

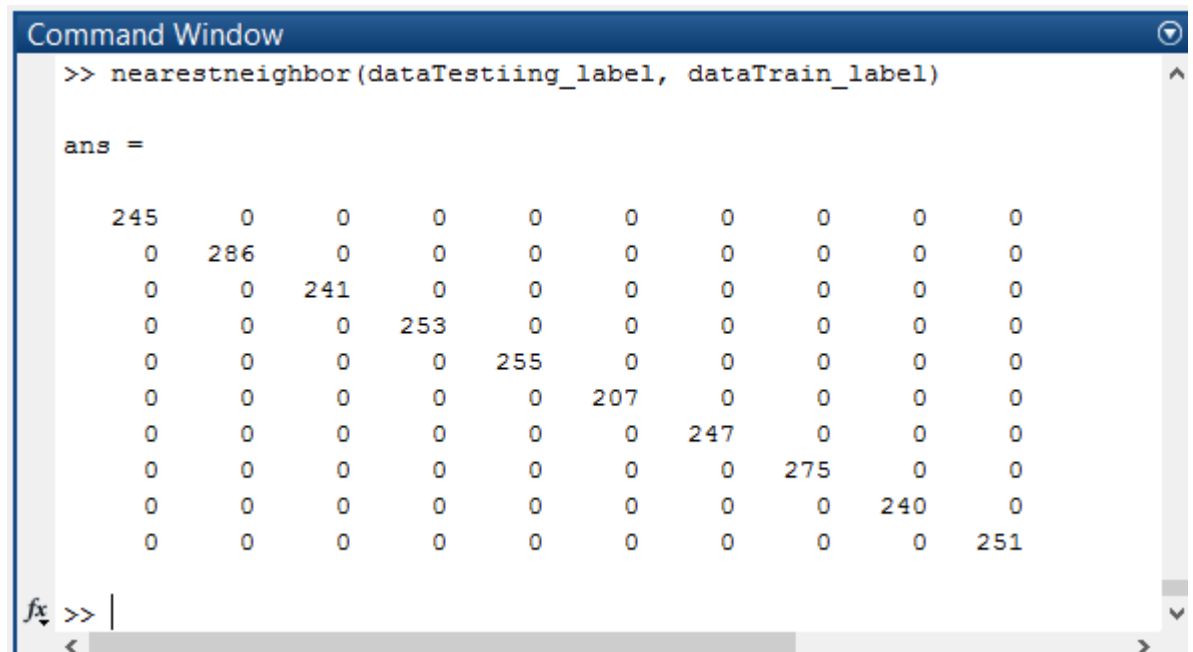
cfsMat = zeros(10);
for i = 1:length(dataTestiing_label);
    vectorEuclidean = [];
```

```
%mencari nilai jarak euclidean dengan membandingkan setiap
data Testing oleh
%data Training
for j = 1:length(dataTrain_label);
    jarakEuclidean = norm(dataTestiing_label(i,1:end-1)-
dataTrain_label(j,1:end-1));
    vectorEuclidean = [vectorEuclidean ; jarakEuclidean
dataTestiing_label(i,end)];
end

[M idx] = min(vectorEuclidean);
posisi = vectorEuclidean(idx(1),2);
%membuat confusion matrix
cfsMat(dataTestiing_label(i,end)+1 , posisi+1) =
cfsMat(dataTestiing_label(i,end)+1 , posisi+1) + 1;
end

hasil = cfsMat;
end
```

- Hasil Nearest Neighbor Classifier



```
Command Window
>> nearestneighbor(dataTestiing_label, dataTrain_label)

ans =

    245     0     0     0     0     0     0     0     0     0
     0    286     0     0     0     0     0     0     0     0
     0     0    241     0     0     0     0     0     0     0
     0     0     0    253     0     0     0     0     0     0
     0     0     0     0    255     0     0     0     0     0
     0     0     0     0     0    207     0     0     0     0
     0     0     0     0     0     0    247     0     0     0
     0     0     0     0     0     0     0    275     0     0
     0     0     0     0     0     0     0     0    240     0
     0     0     0     0     0     0     0     0     0    251
```

- Screenshoot Codingan


```
1 function hasil = nearestneighbor( dataTestiing_label , dataTrain_label )
2
3     cfsMat = zeros(10);
4     for i = 1:length(dataTestiing_label);
5         vectorEuclidean = [];
6
7         %mencari nilai jarak euclidean dengan membandingkan setiap data Testing ole
8         % setiap data Training
9         for j = 1:length(dataTrain_label);
10             jarakEuclidean = norm(dataTestiing_label(i,1:end-1)-dataTrain_label(j,1:end-1));
11             vectorEuclidean = [vectorEuclidean ; jarakEuclidean dataTestiing_label(i,end)];
12         end
13
14         [M idx] = min(vectorEuclidean);
15         posisi = vectorEuclidean(idx(1),2);
16         %membuat confusion matrix
17         cfsMat(dataTestiing_label(i,end)+1 , posisi+1) = cfsMat(dataTestiing_label(i,end)+1 , posisi+1) + 1;
18     end
19
20     hasil = cfsMat;
21 end
22
23
```

- Analisis Program

Fungsi diatas untuk memperoleh confusion matrix dari algoritma nearest neighbor classifier. Nearest neighbor classifier juga menggunakan nilai Euclidean antar data. Tetapi pada algoritma ini pengetesan jarak Euclidean dengan masing-masing data train dan data testing.

E. ~

- Source Code

```
function [ Accuracy Error Precision Recall F1 ] =  
analisisConfusion( hasilCfsMatrix )  
%Accuracy, Recall, Precision dari confusion matrix  
TruePositive = zeros(10,1);  
TrueNegative = zeros(10,1);  
FalsePositive = zeros(10,1);  
FalseNegative = zeros(10,1);  
  
for i=1:10,  
    TruePositive(i) = hasilCfsMatrix(i,i);
```

```
TrueNegative(i) = sum(sum(hasilCfsMatrix)) -  
sum(FalsePositive)-sum(FalseNegative)-TruePositive(i);  
FalsePositive(i) = sum(hasilCfsMatrix(:,i)) -  
TruePositive(i);  
FalseNegative(i) = sum(hasilCfsMatrix(i,:)) -  
TruePositive(i);  
end  
  
Accuracy = (sum(TruePositive) + sum(TrueNegative)) /  
(sum(TruePositive) + sum(TrueNegative) + sum(FalsePositive) +  
sum(FalseNegative));  
Error = (sum(FalsePositive) + sum(FalseNegative)) /  
(sum(TruePositive) + sum(TrueNegative) + sum(FalsePositive) +  
sum(FalseNegative));  
Precision = sum(TruePositive) / (sum(TruePositive) +  
sum(FalsePositive));  
Recall = sum(TruePositive) / (sum(TruePositive) +  
sum(FalseNegative));  
F1 = (2*Precision*Recall) / (Precision+Recall);  
  
end
```

- Hasil analisis confusion matrix

~ Nearest Neighbor

```
acc =  
  
    1  
  
err =  
  
    0  
  
prec =  
  
    1  
  
rec =  
  
    1  
  
f1 =  
  
    1
```

~ Prototype Classifier

```
acc =  
    0.9388  
  
err =  
    0.0612  
  
prec =  
    0.7436  
  
recall =  
    0.7436  
  
f1 =  
    0.7436
```

- Screen shoot codingan

```
1 function [ Accuracy Error Precision Recall F1 ] = analisisConfusion( hasilCfsMa
2 %Accuracy, Recall, Precision dari confusion matrix
3 TruePositive = zeros(10,1);
4 TrueNegative = zeros(10,1);
5 FalsePositive = zeros(10,1);
6 FalseNegative = zeros(10,1);
7
8
9 for i=1:10,
10 TruePositive(i) = hasilCfsMatrix(i,i);
11 TrueNegative(i) = sum(sum(hasilCfsMatrix)) - sum(FalsePositive) - sum(FalseNe
12 FalsePositive(i) = sum(hasilCfsMatrix(:,i)) - TruePositive(i);
13 FalseNegative(i) = sum(hasilCfsMatrix(i,:)) - TruePositive(i);
14 end
15
16 Accuracy = (sum(TruePositive) + sum(TrueNegative)) / (sum(TruePositive) + s
17 Error = (sum(FalsePositive) + sum(FalseNegative)) / (sum(TruePositive) + sum
18 Precision = sum(TruePositive) / (sum(TruePositive) + sum(FalsePositive));
19 Recall = sum(TruePositive) / (sum(TruePositive) + sum(FalseNegative));
20 F1 = (2*Precision*Recall) / (Precision+Recall);
21
22 end
23
24
```

- Analisis Program

Fungsi diatas untuk mencari f1 score dari confusion matrix. Data confusion matrix dapat digunakan untuk mencari nilai jumlah true positive, jumlah true negative, jumlah false positive, dan jumlah false negative.

Nearest Neighbor Classifier memberikan hasil terbaik dengan 1 f1-score dari rentang 0-1. Tidak ada miss untuk metode ini.

Digit 5 paling banyak salah diklasifikasikan karena pixel yang menyebar dan tidak detail untuk gambar prototype pada digit ke-5