SATRYA BUDI PRATAMA
1301154428

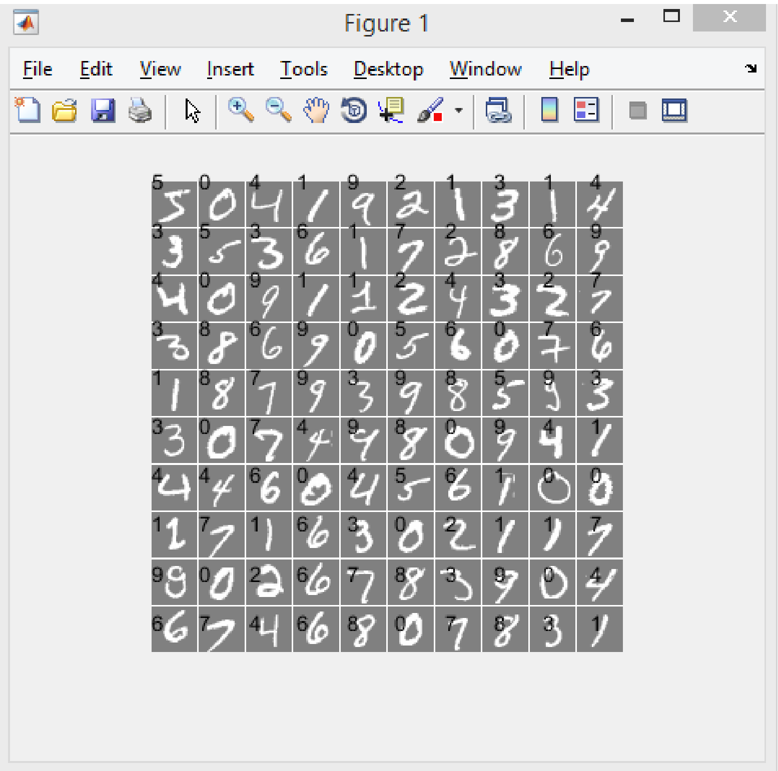
7.

A. ~

- Source Code

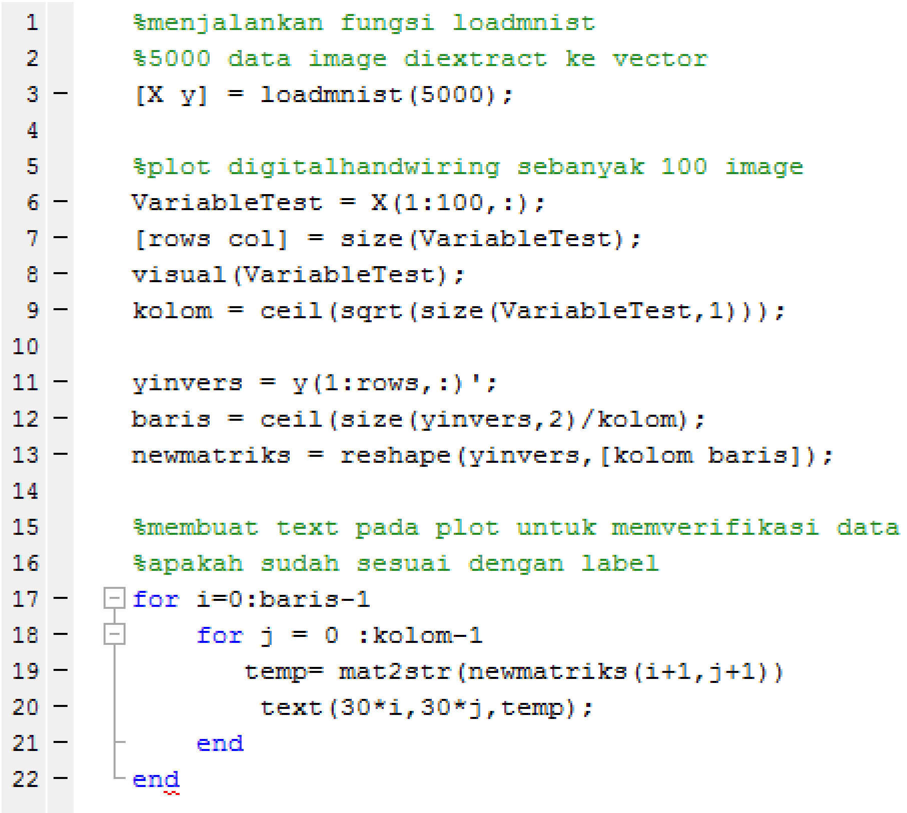
```
%menjalankan fungsi loadmnist
%5000 data image diextract ke vector
[X y] = loadmnist(5000);

%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



SATRYA BUDI PRATAMA
1301154428

- Screenshoot codingan



```
%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
```

- 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. ~

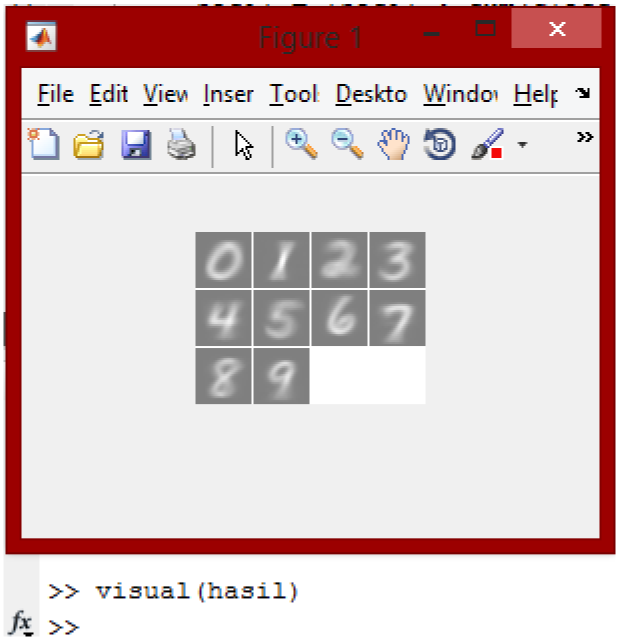
- Source Code & Penjelasan

```
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
```

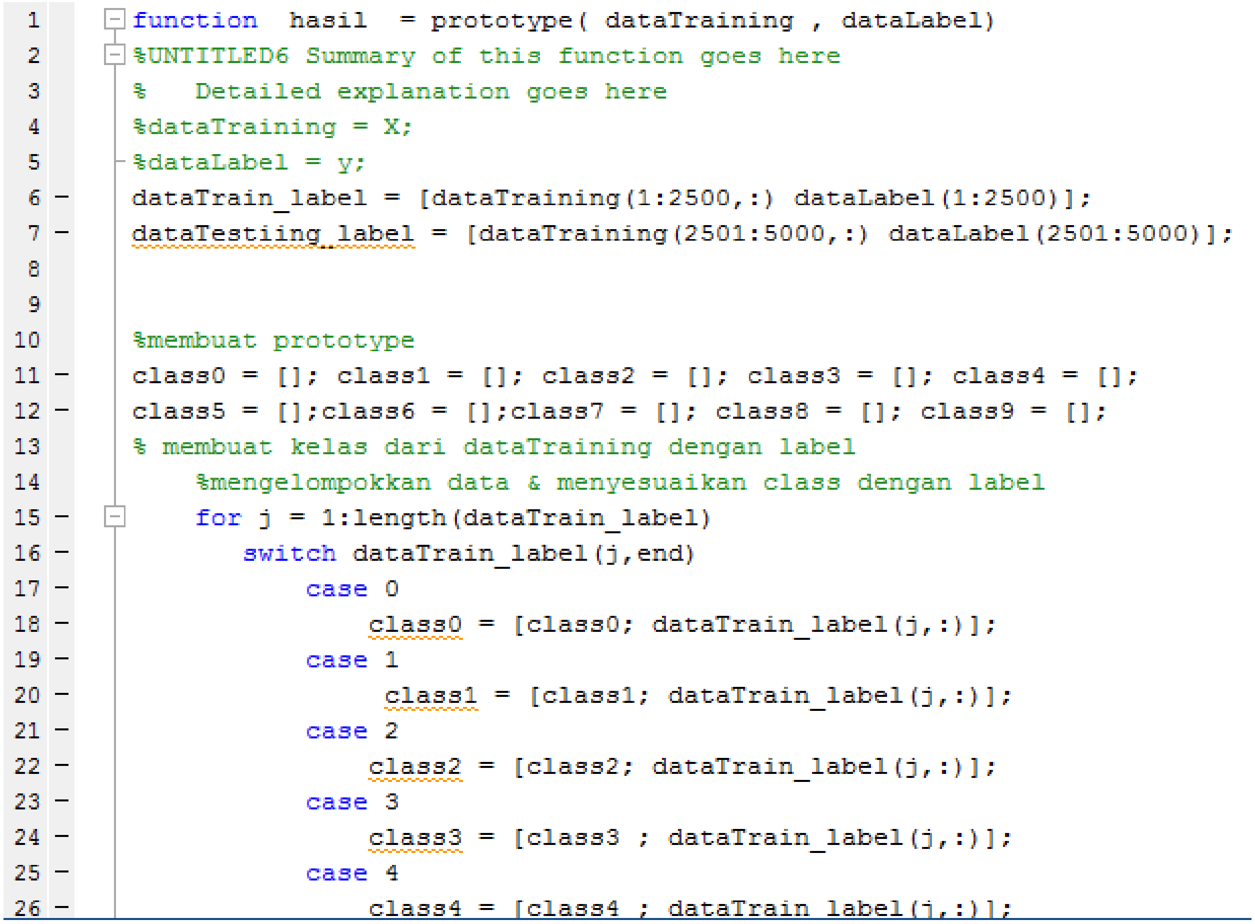
- ScreenShoot hasil

SATRYA BUDI PRATAMA
1301154428

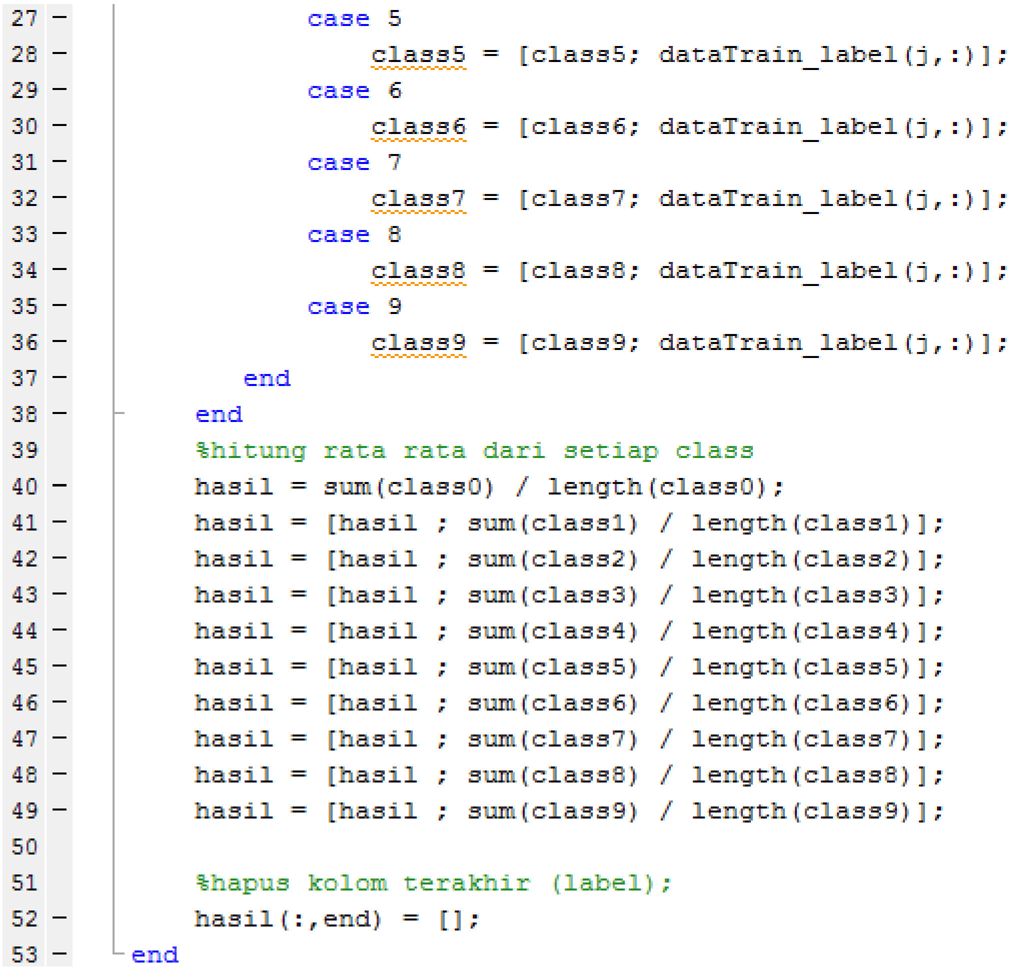


```
>> visual(hasil)
>>
```

- ScreenShoot codingan



```
function  hasil  = 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
```

- Analisis Program

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

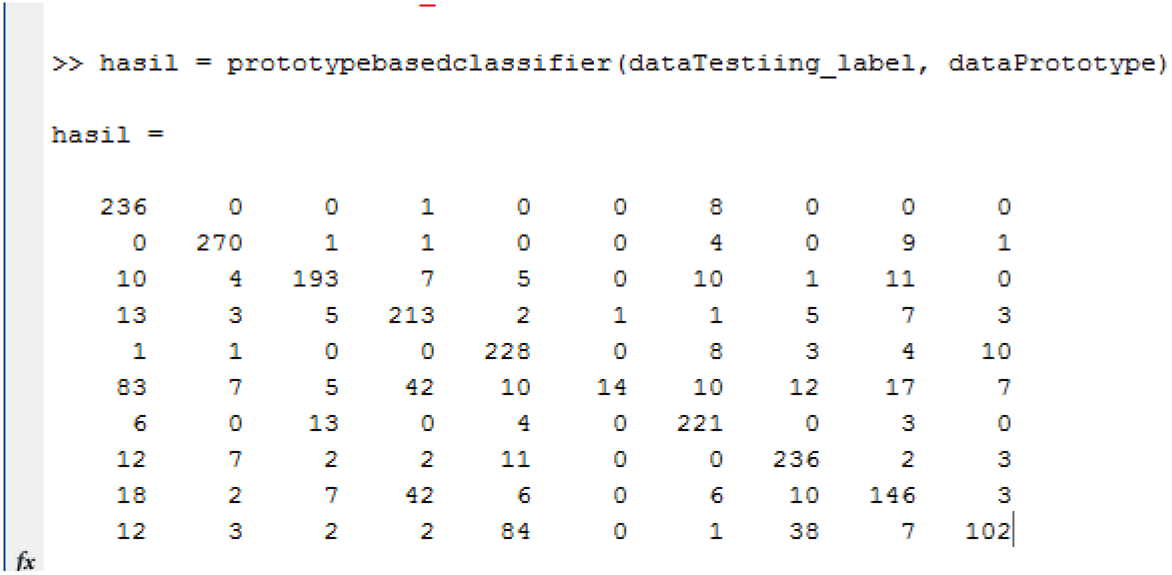
C. ~

- Source Code & Penjelasan

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

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

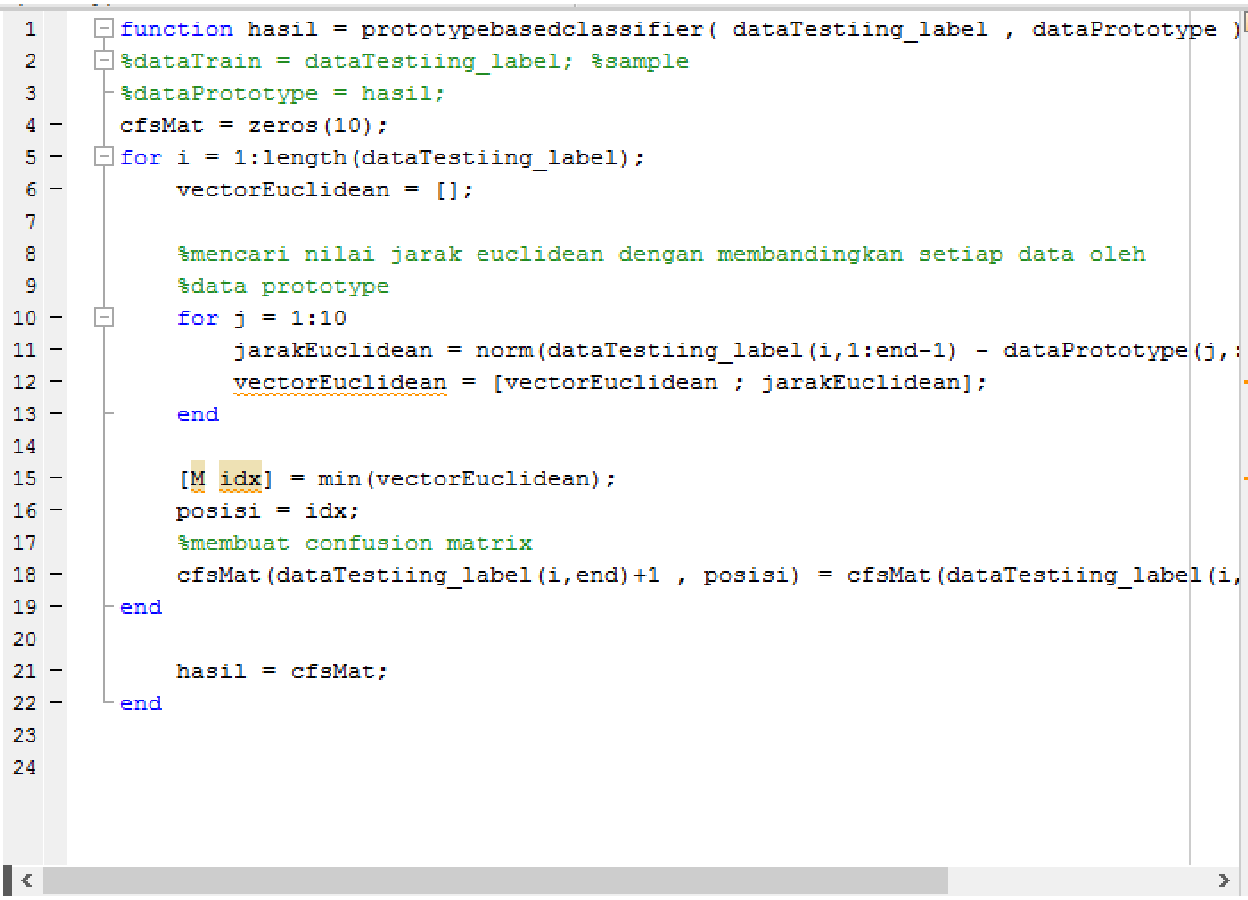
```
    %data prototype
    for j = 1:10
        jarakEuclidean = norm(dataTestiing_label(i,1:end-1) - 
dataPrototype(j,:));
        vectorEuclidean = [vectorEuclidean ; jarakEuclidean];
    end
    
    [M idx] = min(vectorEuclidean);
    posisi = idx;
    %membuat confusion matrix
    cfsMat(dataTestiing_label(i,end)+1 , 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



```
function hasil = prototypebasedclassifier( dataTestiing_label , dataPrototype )
%dataTrain = dataTestiing_label; %sample
%dataPrototype = hasil;
cfsMat = zeros(10);
for i = 1:length(dataTestiing_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

    hasil = cfsMat;
end
```

- Analisis Program

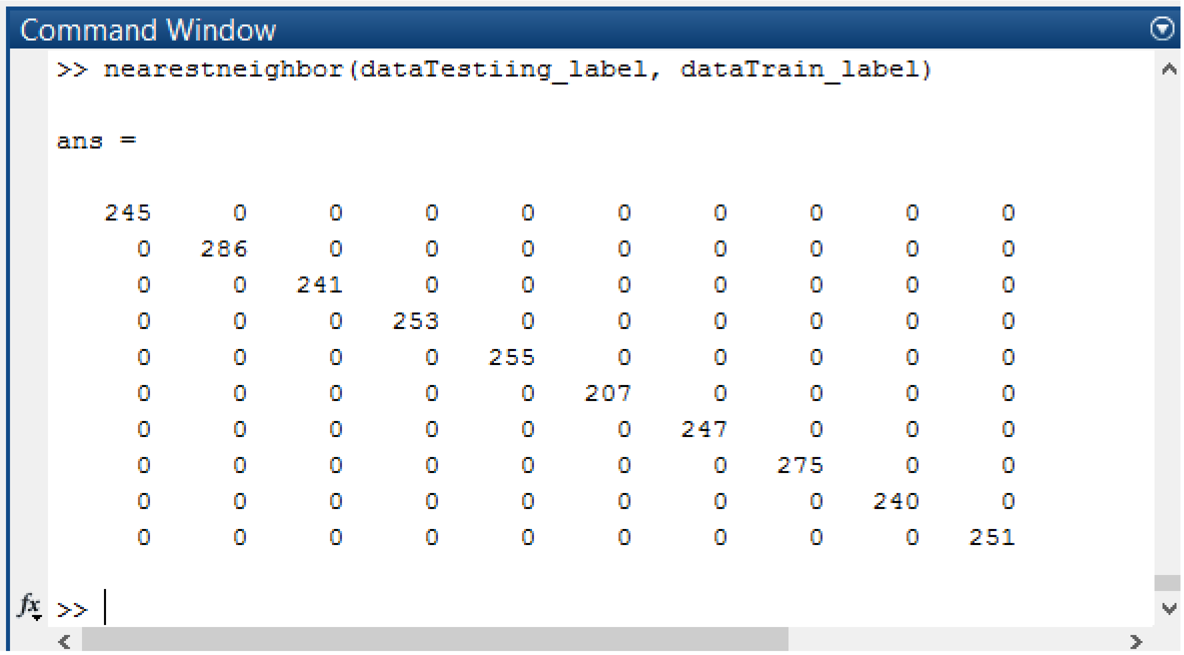
  Fungsi diatas untuk memperoleh hasil confusion matrix yang berisi banyaknya data yang sesuai dengan classnya ataupun yang tidak sesuai dengan classnya. Untuk memperoleh confusion matrix dari setiap class, terlebih dahulu menghitung Euclidean antara seluruh data testing dengan setiap data prototype. Kemudian 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

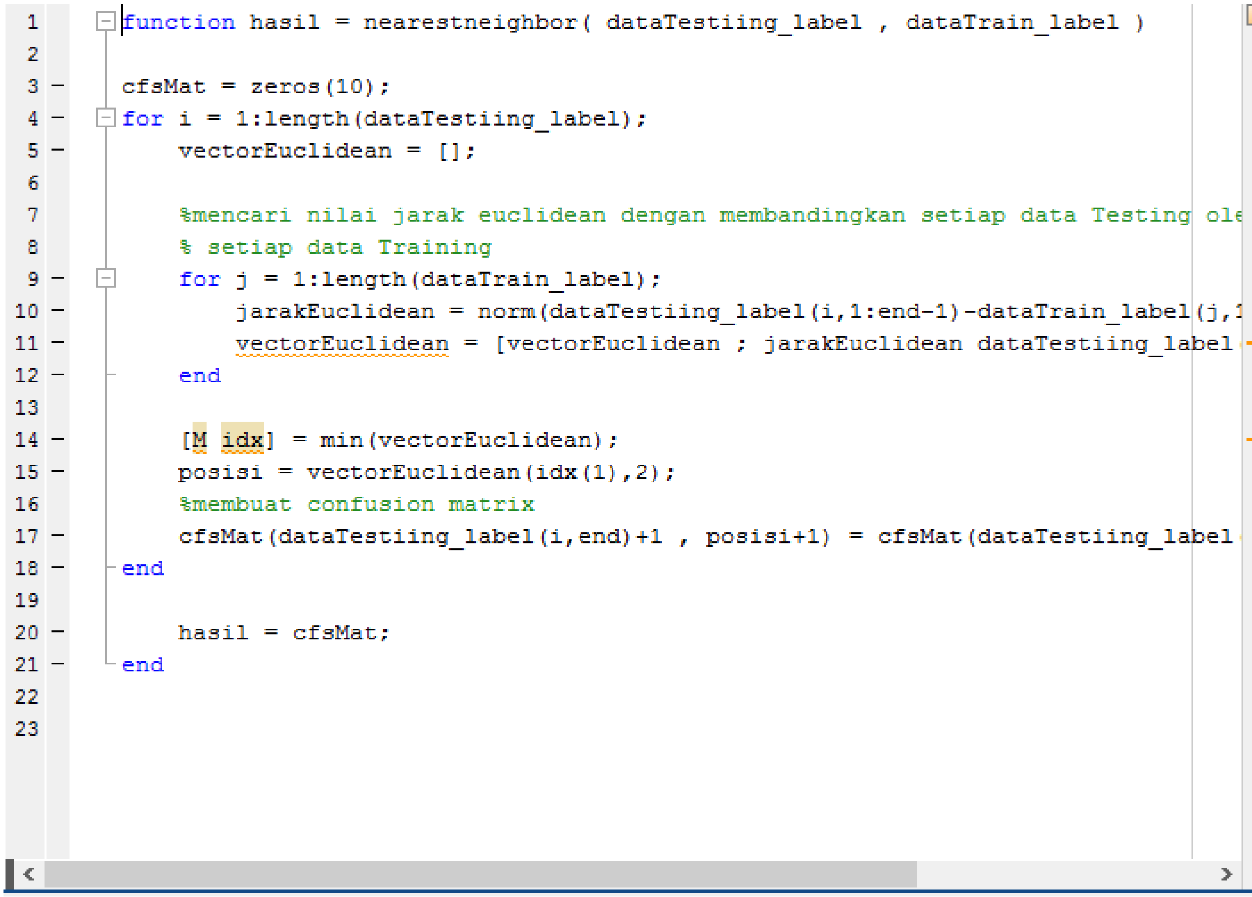
```
function hasil = nearestneighbor( dataTestiing_label ,
dataTrain_label )

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



```
>> 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

fx >> 
```

- Screenshoot Codingan



```
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 ole
    % setiap data Training
    for j = 1:length(dataTrain_label);
        jarakEuclidean = norm(dataTestiing_label(i,1:end-1)-dataTrain_label(j,1
        vectorEuclidean = [vectorEuclidean ; jarakEuclidean dataTestiing_label
    end

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

    hasil = cfsMat;
end
```

- Analysis 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. ~

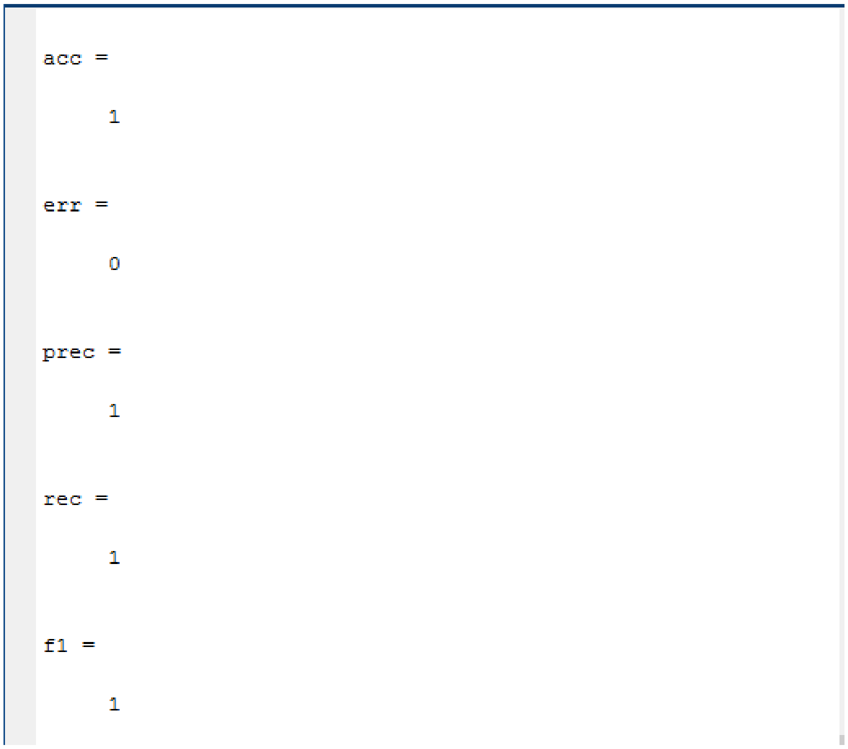
- Source Code

```
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

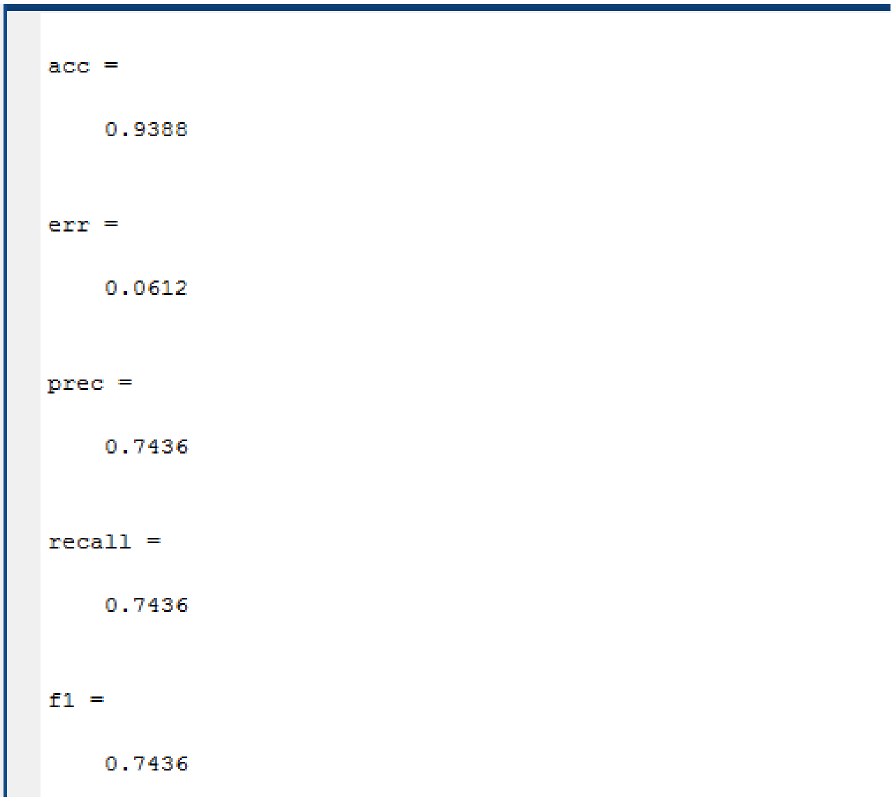
~ Nearest Neighbor



```
acc =

     1


err =

     0


prec =

     1


rec =

     1


f1 =

     1
```

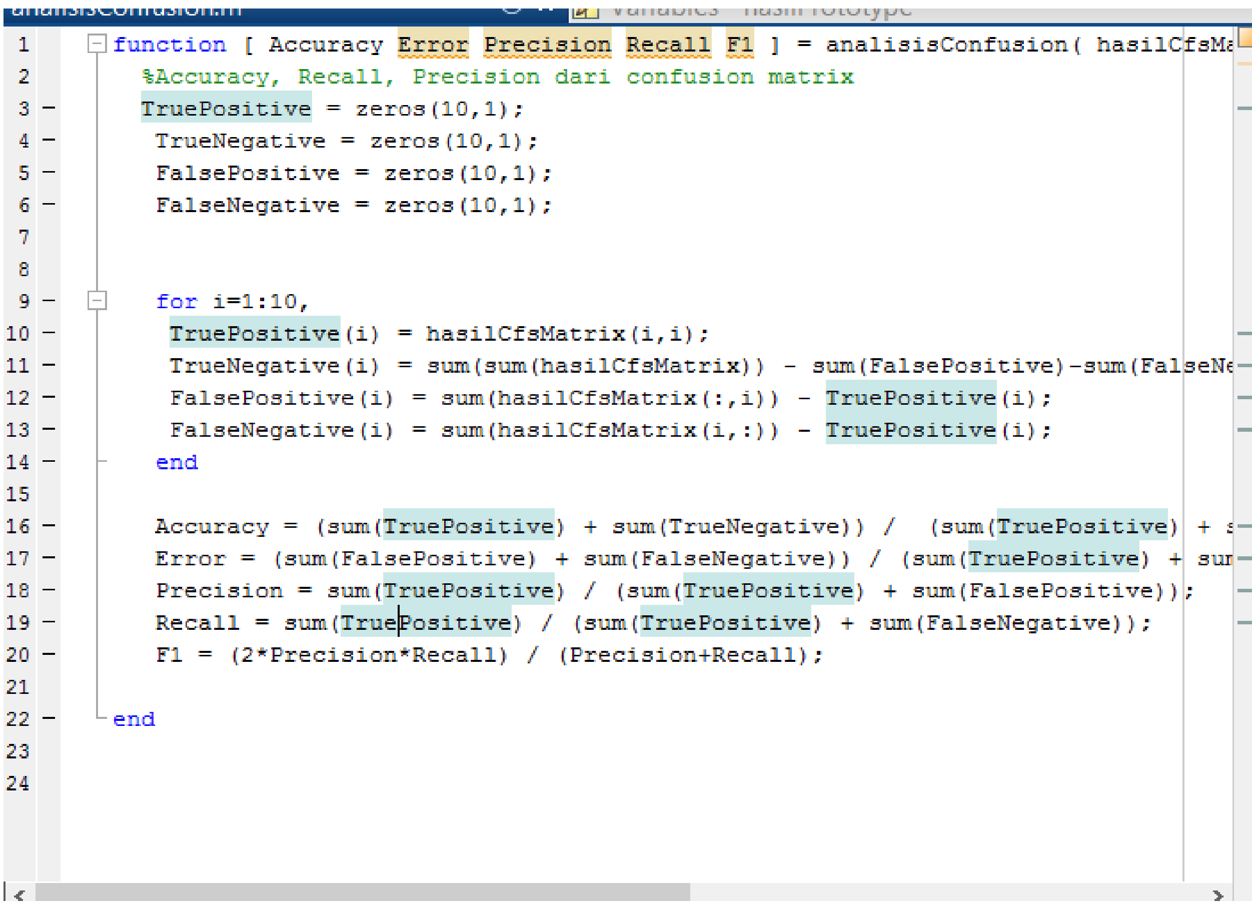
~ Prototype Classfier

SATRYA BUDI PRATAMA
1301154428



```
acc =

    0.9388


err =

    0.0612


prec =

    0.7436


recall =

    0.7436


f1 =

    0.7436
```

- Screen shoot codingan



```
function [ Accuracy Error Precision Recall F1 ] = analisisConfusion( hasilCfsMa
  %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(FalseNe
    FalsePositive(i) = sum(hasilCfsMatrix(:,i)) - TruePositive(i);
    FalseNegative(i) = sum(hasilCfsMatrix(i,:)) - TruePositive(i);
   end

   Accuracy = (sum(TruePositive) + sum(TrueNegative)) /  (sum(TruePositive) + s
   Error = (sum(FalsePositive) + sum(FalseNegative)) / (sum(TruePositive) + su
   Precision = sum(TruePositive) / (sum(TruePositive) + sum(FalsePositive));
   Recall = sum(TruePositive) / (sum(TruePositive) + sum(FalseNegative));
   F1 = (2*Precision*Recall) / (Precision+Recall);

end
```

- 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