```
%Reading the data from the CSV file
data = csvread('banknote_authentication.csv',1,0)
```

```
data = 1372×5
   2.783100000000000  10.979600000000000  -3.557000000000000  -4.403900000000000 ···
  -0.360250000000000  -4.449000000000000   2.106700000000000   0.943080000000000
   0.163580000000000  -3.358400000000000   1.374900000000000   1.356900000000000
   4.119700000000000  -2.795600000000000   2.070700000000000   0.674120000000000
  -1.651400000000000  -8.498500000000000   9.112200000000000   1.237900000000000
  -7.042100000000000   9.199999999999999   0.259330000000000  -4.683200000000000
   1.866400000000000   7.776300000000000  -0.238490000000000  -2.963400000000000
   3.884600000000000  -3.033600000000000   2.533400000000000   0.202140000000000
   3.203200000000000   5.758800000000000  -0.753450000000000  -0.612510000000000
   0.110320000000000   1.974100000000000  -3.366800000000000  -0.652590000000000
      ⋮
      ⋮
```

```
%Building the Training and Testing sets
%The training set is 70% of the data
% The testing set is 30% of the data
Training_Set = data(1:961,:)
```

```
Training_Set = 961×5
   2.783100000000000  10.979600000000000  -3.557000000000000  -4.403900000000000 ···
  -0.360250000000000  -4.449000000000000   2.106700000000000   0.943080000000000
   0.163580000000000  -3.358400000000000   1.374900000000000   1.356900000000000
   4.119700000000000  -2.795600000000000   2.070700000000000   0.674120000000000
  -1.651400000000000  -8.498500000000000   9.112200000000000   1.237900000000000
  -7.042100000000000   9.199999999999999   0.259330000000000  -4.683200000000000
   1.866400000000000   7.776300000000000  -0.238490000000000  -2.963400000000000
   3.884600000000000  -3.033600000000000   2.533400000000000   0.202140000000000
   3.203200000000000   5.758800000000000  -0.753450000000000  -0.612510000000000
   0.110320000000000   1.974100000000000  -3.366800000000000  -0.652590000000000
      ⋮
      ⋮
```

```
Testing_Set=data(962:1372,:)
```

```
Testing_Set = 411×5
  -1.525200000000000  -6.253400000000000   5.352400000000000   0.599120000000000 ···
   0.813560000000000   9.156599999999999  -2.149200000000000  -4.181400000000000
  -0.508160000000000   2.868000000000000  -1.810800000000000  -2.261200000000000
   0.322300000000000  -0.898080000000000   8.088300000000000   0.692220000000000
   1.356200000000000   3.213600000000000   4.346500000000000   0.786620000000000
   0.273310000000000   4.877300000000000  -4.919400000000000  -5.819800000000000
  -2.575400000000000  -5.657400000000000   6.103000000000000   0.652140000000000
  -0.280150000000000   3.072900000000000  -3.385700000000000  -2.915500000000000
  -0.117160000000000   0.604220000000000  -0.385870000000000  -0.059065000000000
  -0.826010000000000   2.961100000000000  -1.286400000000000  -1.464700000000000
      ⋮
      ⋮
```

```
%Taking transpose of datasets
Training_Set_Transpose=Training_Set';
Testing_Set_Transpose=Testing_Set';
```

```
%Normalizing the Training dataset and Preparing the X_Train_1
X_Train_1 = zeros(4,961);
m = zeros(1,4);
v = zeros(1,4);
for i = 1:4
 xi = Training_Set_Transpose(i+1,:);
 m(i) = mean(xi);
 v(i) = sqrt(var(xi));
 X_Train_1(i,:) = (xi- m(i))/v(i);
end
```

```
%Normalizing Testing Dataset and Preparing the X_Test_1
X_Test_1 = zeros(4,411);
for i = 1:4
    xi = Testing_Set_Transpose(i+1,:);
    X_Test_1(i,:) = (xi - m(i))/v(i);
end
```

```
%Preparing the Label/ Target Vector
y_train = Training_Set_Transpose(5,:);
y_test = Testing_Set_Transpose(5,:);
```

```
% Data post Normalization
X_train = [X_Train_1; y_train];
X_test = [X_Test_1; y_test];
w0 = zeros(5,1);
```

```
% Expermineting with different values of mu and K
%mu = 0; K = 10;
%mu = 0.1; K = 10;
mu = 0; K = 30;
%mu = 0.05; K = 30;
```

```
% Calculating the Optimized Weight in Gradient Descent
ws_1 = grad_desc('f_wdbc','g_wdbc',w0,X_train,mu,K);
```

```
% Evaluating model performance over the Testing Dataset
p_test = find(y_test==1);
n_test = find(y_test==0);
prediction = zeros(1,412);

for i = 1:length(X_Test_1)
    xi = [X_Test_1(:,i);1];
    yi = sign(ws_1'*xi);
    if yi==1
        prediction(1,i) = 1;
    else
        prediction(2,i) = 1;

    end
end




% Building the CONFUSION_MATRIX_test
C1_test = sum(prediction(:,p_test),2);
C2_test = sum(prediction(:,n_test),2);




% Calculating the ACCCURACY_test
sum_of_elements_test=C2_test(2)+C1_test(1)+C2_test(1)+C1_test(2)
```

```
sum_of_elements_test =
   411
```

```
tp_tn_sum_test=C2_test(2)+C1_test(1)
```

```
tp_tn_sum_test =
   348
```

```
%Calculating the PRECISION_test
tp_test=C2_test(2)
```

```
tp_test =
   166
```

```
tp_fp_sum_test=C2_test(1)+C2_test(2)
```

```
tp_fp_sum_test =
   229
```

```
%Calculating the RECALL_test
tp_fn_sum_test=C1_test(2)+C2_test(2)
```

```
tp_fn_sum_test =
   166
```

```
%Calculating the F1_SCORE_test
p_r_sum_test=PRECISION_test+RECALL_test
```

```
p_r_sum_test =
   1.716157205240175
```

```
p_r_multiply_test=PRECISION_test*RECALL_test
```

```
p_r_multiply_test =
   0.716157205240175
```

```
CONFUSION_MATRIX_test = [C1_test C2_test]
```

```
CONFUSION_MATRIX_test = 2×2
   182    63
     0   166
```

```
ACCCURACY_test= tp_tn_sum_test/sum_of_elements_test
```

```
ACCCURACY_test =
   0.846715328467153
```

```
PRECISION_test=tp_test/tp_fp_sum_test
```

```
PRECISION_test =
   0.724890829694323
```

```
RECALL_test=tp_test/tp_fn_sum_test
```

```
RECALL_test =
     1
```

```
F1_SCORE_test=2*p_r_multiply_test/p_r_sum_test
```

```
F1_SCORE_test =
   0.834605597964377
```

```
% Evaluating model performance over the Training Dataset
p_train = find(y_train==1);
n_train = find(y_train==0);
prediction_train = zeros(1,412);

for i = 1:length(X_Train_1)
    xi = [X_Train_1(:,i);1];
    yi = sign(ws_1'*xi);
    if yi==1
        prediction_train(1,i) = 1;
    else
        prediction_train(2,i) = 1;

    end
end
```

```
% Building the CONFUSION_MATRIX_train
C1_train = sum(prediction_train(:,p_train),2);
C2_train = sum(prediction_train(:,n_train),2);
```

```
% Calculating the ACCCURACY_train
sum_of_elements_train=C2_train(2)+C1_train(1)+C2_train(1)+C1_train(2)
```

```
sum_of_elements_train =
   961
```

```
tp_tn_sum_train=C2_train(2)+C1_train(1)
```

```
tp_tn_sum_train =
   827
```

```
%Calculating the PRECISION_train
tp_train=C2_train(2)
```

```
tp_train =
   399
```

```
tp_fp_sum_train=C2_train(1)+C2_train(2)
```

```
tp_fp_sum_train =
   533
```

```
%Calculating the RECALL_train
tp_fn_sum_train=C1_train(2)+C2_train(2)
```

```
tp_fn_sum_train =
   399
```

```
%Calculating the F1_SCORE_train
p_r_sum_train=PRECISION_train+RECALL_train
```

```
p_r_sum_train =
   1.737335834896811
```

```
p_r_multiply_train=PRECISION_train*RECALL_train
```

```
p_r_multiply_train =
   0.737335834896811
```

```
CONFUSION_MATRIX_train = [C1_train C2_train]
```

```
CONFUSION_MATRIX_train = 2×2
   428   134
     0   399
```

```
ACCCURACY_train= tp_tn_sum_train/sum_of_elements_train
```

```
ACCCURACY_train =
   0.860561914672216
```

```
PRECISION_train=tp_train/tp_fp_sum_train
```

```
PRECISION_train =
   0.748592870544090
```

```
RECALL_train=tp_train/tp_fn_sum_train
```

```
RECALL_train =
     1
```

```
F1_SCORE_train=2*p_r_multiply_train/p_r_sum_train
```

```
F1_SCORE_train =
   0.848812095032397
```