Homework 5

SVM for Shaft Health Assessment

Shashank Iyengar ***Johann Koshy*** ***Ashwin Kumat*** ***Ketan Shah***

The objective of this exercise was to use SVM for assessing health rotating shaft anchored on a testbed with training and testing data sets. Training data is classified as healthy (balanced shaft) and faulty (unbalance type 1 and unbalance type 2).

A. Briefing analysis procedure

1. Read acceleration values from given train/test data.
2. Extract features like RMS, Peak to Peak, Kurtosis and Skewness features from the time domain training and testing data.
3. Perform FFT for each data set to extract amplitude feature at 1st and 2nd harmonic frequency.

Parameters for FFT:

```
Fs = 2560;          % Sampling frequency
dt = 1/Fs;          % Time step
Ntime = 38400;      % Number of data points
Ttotal = 15;        % Total time
df = 1/Ttotal;      % Fundamental frequency
```

Amplitudes of peaks corresponding to 20Hz and 40Hz are extracted and stored in a row vector of size 1x20 each for healthy and faulty data. Similarly, features are extracted for the test data.

4. Form a feature matrix for training and testing data with number of columns indicating number of features extracted and number of rows indicating number of data samples.
5. Data was normalized with mean= 0 and variance =1 for performing PCA for dimensionality reduction.
6. PCA was performed on RMS, Peak to Peak, amplitude 1x , 2x and the eigen values were plotted to analyse and to find the principal components.
7. These Principal components were then used to train the SVM model and the test data features were projected in PCA space by multiplying with eigen vector and were passed through this model to obtain the prediction.
8. Confusion matrix was then created to analyse the prediction result and hit rate was calculated from this matrix to find the accuracy of the model.

B. Figures showing assessment result

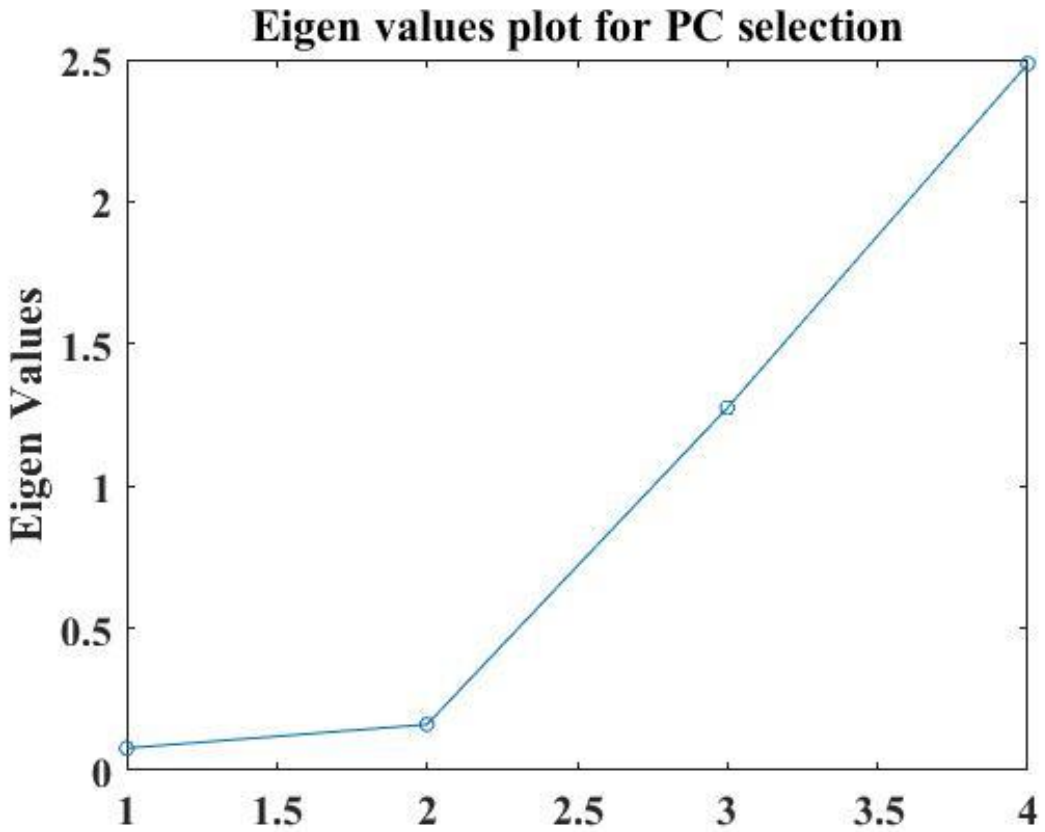

Figure 1: Eigen value plot for selection of Principle component (Selecting PC 3 and PC 4)

Confusion Matrix					
		Actual Data			
		Healthy	Unbalance 1	Unbalance 2	
Predicted Data	Healthy	10	0	0	10
	Unbalance 1	0	10	0	10
	Unbalance 2	0	0	10	10
		10	10	10	30

Table 1: Confusion Matrix

Sample number	Analysis	File number	Analysis	File number	Analysis
1	Unbalance 1	11	Unbalance 1	21	Unbalance 2
2	Unbalance 1	12	Unbalance 1	22	Unbalance 2
3	Healthy	13	Unbalance 1	23	Unbalance 2
4	Healthy	14	No result	24	Unbalance 2
5	Unbalance 1	15	No result	25	Unbalance2
6	Unbalance 1	16	Unbalance 1	26	Unbalance 2
7	Unbalance 1	17	Unbalance 1	27	Unbalance 2
8	Healthy	18	Unbalance 1	28	No result
9	Healthy	19	Unbalance1	29	No result
10	No result	20	Unbalance 2	30	Unbalance 2

Table 2: Classification by SOM

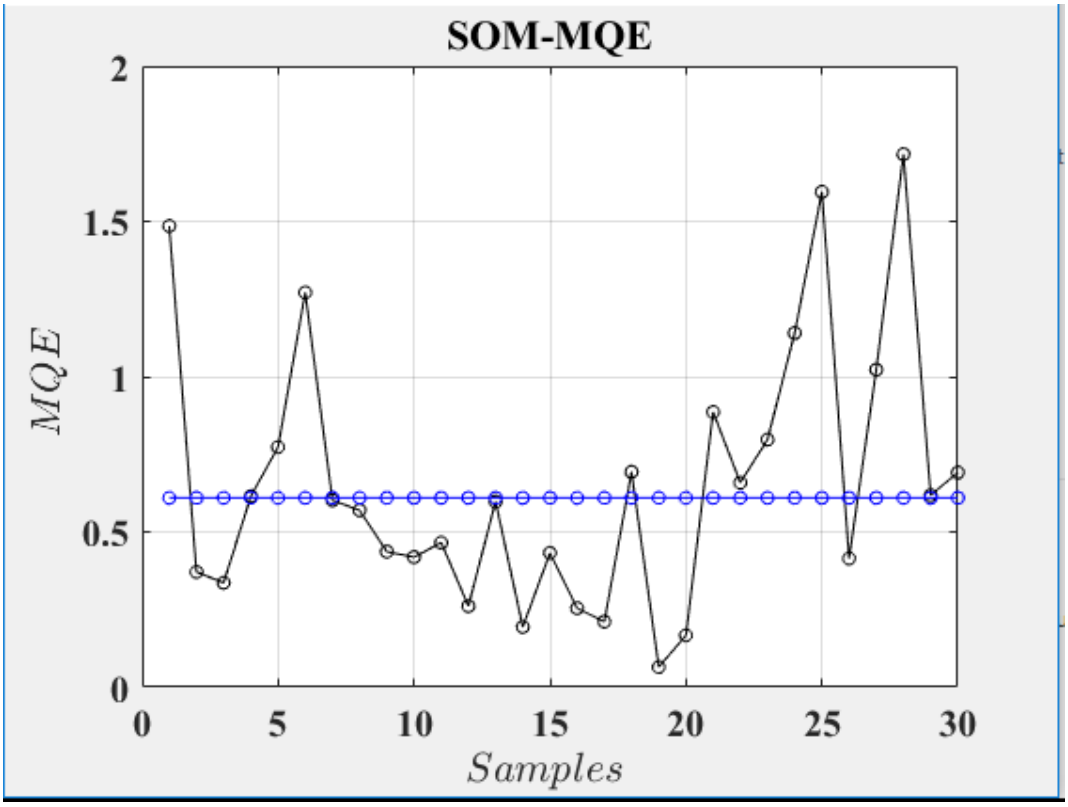

Figure 2: MQE for test samples. (Blue line indicates baseline)

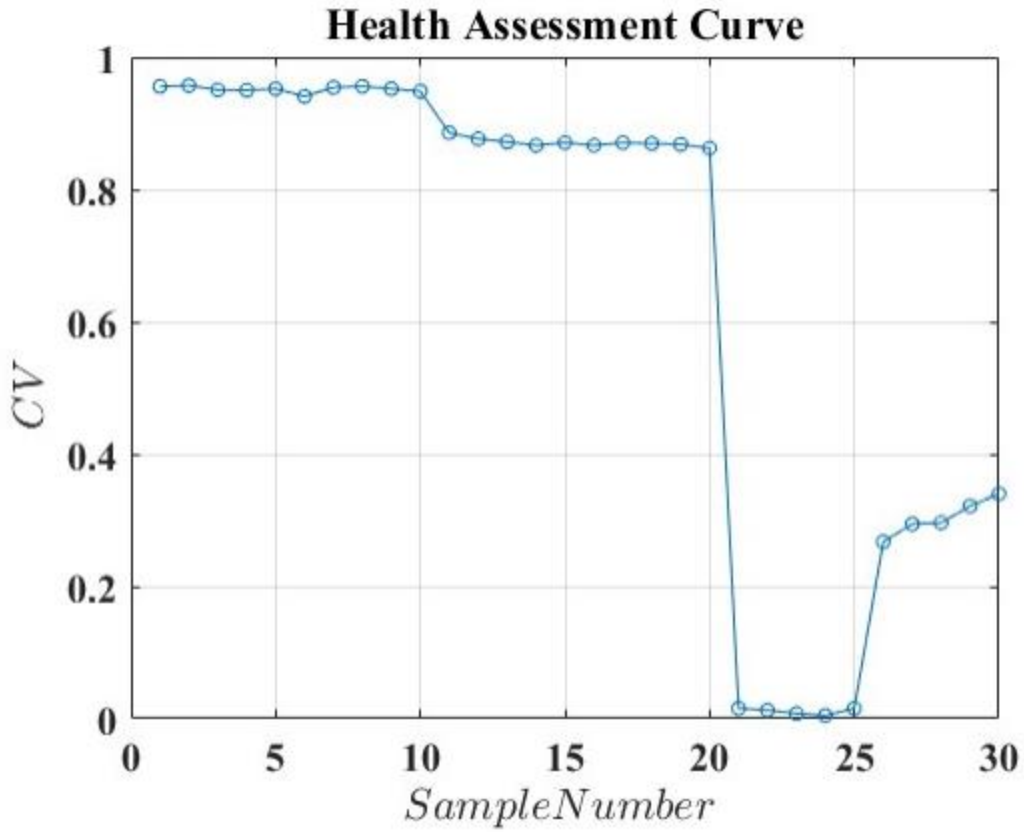

Figure 3: Logistic Regression result from Homework 3

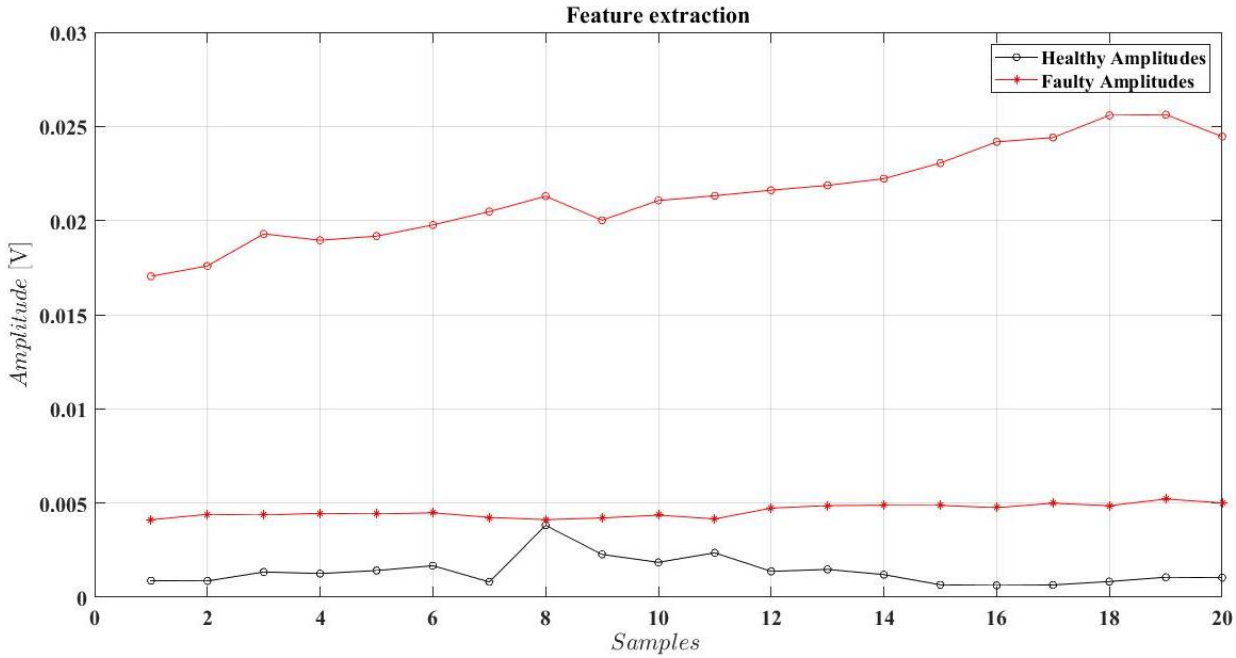

Figure 4: Amplitude features for Training data (Faulty – Unbalance 1, Unbalance 2 and Healthy)

C. **Analysis of the Results**

1. PCA was performed on RMS, Peak to Peak, amplitude 1x, 2x as including more features in PCA for dimensionality reduction reduced the accuracy of predication of SVM model.
2. From Figure 1 we can easily see that there is a steep increase in eigen value for 3rd column of PCA matrix and thus last two columns were selected as principal components.
3. The SVM model was accurately able to classify Healthy, unbalance 1 and unbalance 2 data which can be inferred from the confusion matrix result from Table 1.
4. **The accuracy of the trained model was found to be 100%.**

D. Results comparison between Logistic regression, SOM, SOM-MQE and SVM

1. In the homework 3 data for logistic regression the difference between feature metric of healthy and faulty data was quite significant as compared to that between unbalance 1 and healthy data in homework 4. Thus, it is difficult to classify the data between unbalance 1 and healthy.
2. SOM is based on neural network and we need large amount of data to train the model to get better result compared to Logistic regression model. In this homework, the data may have been insufficient for obtaining better classification between healthy, unbalance 1 and unbalance 2 class.
3. In case of SVM model, extracting 2x frequency amplitude significantly improved the accuracy compared to other models. This model was able to **accurately** separate Unbalance 1 and Healthy data which was not observed in case of LR and SOM models.
4. Thus, SVM generated **best** results among all previously used models.

E. Matlab Scripts

```
%% Homework 5 - Shaft Health Assessment - SVM
%% Group 1 - Shashank Iyengar, Johann Koshy, Ashwin Kumat, Ketan Shah

close all
clear all
clc
set(0,'DefaultAxesFontName', 'Times New Roman')
set(0,'DefaultAxesFontSize', 15)
set(0,'defaultlinelinewidth',.5)
set(0,'DefaultLineMarkerSize', 5)
set(0,'defaultAxesFontWeight','bold')

%% Data Acquisition
% Training - Healthy Data
%testfiledir =
'C:\Users\johan\Desktop\UC_Spring2019\Big_data\HW5\Training\Healthy';
```

```
%testfiledir = 'C:\Users\kumatad\OneDrive - University of Cincinnati\Spring
19 courses\Industrial Big Data\Assignments - IBD\Assignment
5\Training(1)\Training\Healthy';
testfiledir = 'C:\Users\Ashwini\OneDrive - University of Cincinnati\Spring 19
courses\Industrial Big Data\Assignments - IBD\Assignment
5\Training(1)\Training\Healthy';

matfiles = dir(fullfile(testfiledir, '*.txt'));
nfiles = length(matfiles);
data  = cell(nfiles);
for i=1:nfiles
    data{i} = dlmread(fullfile(testfiledir, matfiles(i).name), ' ', 5, 0);
end

% Splitting the data array to parts
train_healthy=[];
for i=1:nfiles
    train_healthy(i,:)=cell2mat(data(i,1)); % Healthy data
end

% Training - Faulty Data Unbalance 1
%testfiledir =
'C:\Users\johan\Desktop\UC_Spring2019\Big_data\HW5\Training\Faulty\Unbalance_
1';
%testfiledir = 'C:\Users\kumatad\OneDrive - University of Cincinnati\Spring
19 courses\Industrial Big Data\Assignments - IBD\Assignment
5\Training(1)\Training\Faulty\Unbalance 1';
testfiledir = 'C:\Users\Ashwini\OneDrive - University of Cincinnati\Spring 19
courses\Industrial Big Data\Assignments - IBD\Assignment
5\Training(1)\Training\Faulty\Unbalance 1';
matfiles = dir(fullfile(testfiledir, '*.txt'));
nfiles = length(matfiles);
data  = cell(nfiles);
for i=1:nfiles
    data{i} = dlmread(fullfile(testfiledir, matfiles(i).name), ' ', 5, 0);
end

% Splitting the data array to parts
train_faulty_ub1=[];
for i=1:nfiles
    train_faulty_ub1(i,:)=cell2mat(data(i,1)); % Unbalance 1 data
end

% Training - Faulty Data Unbalance 2
%testfiledir =
'C:\Users\johan\Desktop\UC_Spring2019\Big_data\HW5\Training\Faulty\Unbalance_
2';
%testfiledir = 'C:\Users\kumatad\OneDrive - University of Cincinnati\Spring
19 courses\Industrial Big Data\Assignments - IBD\Assignment
5\Training(1)\Training\Faulty\Unbalance 2';
testfiledir = 'C:\Users\Ashwini\OneDrive - University of Cincinnati\Spring 19
courses\Industrial Big Data\Assignments - IBD\Assignment
5\Training(1)\Training\Faulty\Unbalance 2';
matfiles = dir(fullfile(testfiledir, '*.txt'));
nfiles = length(matfiles);
data  = cell(nfiles);
```

```
for i=1:nfiles
    data{i} = dlmread(fullfile(testfiledir, matfiles(i).name), ' ', 5, 0);
end

% Splitting the data array to parts
train_faulty_ub2=[];
for i=1:nfiles
    train_faulty_ub2(i,:)=cell2mat(data(i,1)); % Unbalance 2 data
end

% Testing Data (30 Sets)
%testfiledir = 'C:\Users\johan\Desktop\UC_Spring2019\Big_data\HW5\Testing';
%testfiledir = 'C:\Users\kumatad\OneDrive - University of Cincinnati\Spring
19 courses\Industrial Big Data\Assignments - IBD\Assignment
5\Testing(1)\Testing';
testfiledir = 'C:\Users\Ashwini\OneDrive - University of Cincinnati\Spring 19
courses\Industrial Big Data\Assignments - IBD\Assignment
5\Testing(1)\Testing';
matfiles = dir(fullfile(testfiledir, '*.txt'));
nfiles = length(matfiles);
data  = cell(nfiles);
for i = 1 : nfiles
    data{i} = dlmread(fullfile(testfiledir, matfiles(i).name), ' ', 5, 0);
end

% Splitting the data array to parts
test_data=[];
for i=1:nfiles
    test_data(i,:)=cell2mat(data(i,1));
end



%% Feature extraction Time Domain %%
%%%% RMS - Peak to Peak - Skewness - Kurtosis Training
for i=1:20

    rms_healthy(i) = rms(train_healthy(i,:));
    rms_faulty_ub1(i) = rms(train_faulty_ub1(i,:));
    rms_faulty_ub2(i) = rms(train_faulty_ub2(i,:));

    p2p_healthy(i) = peak2peak(train_healthy(i,:));
    p2p_faulty_ub1(i) = peak2peak(train_faulty_ub1(i,:));
    p2p_faulty_ub2(i) = peak2peak(train_faulty_ub2(i,:));

    skewness_healthy(i) = skewness(train_healthy(i,:));
    skewness_faulty_ub1(i) = skewness(train_faulty_ub1(i,:));
    skewness_faulty_ub2(i) = skewness(train_faulty_ub2(i,:));

    kurtosis_healthy(i) = kurtosis(train_healthy(i,:));
    kurtosis_faulty_ub1(i) = kurtosis(train_faulty_ub1(i,:));
    kurtosis_faulty_ub2(i) = kurtosis(train_faulty_ub2(i,:));

end
%%%% RMS Testing
```

```
for i=1:30
    rms_test(i) = rms(test_data(i,:));
    p2p_test(i) = peak2peak(test_data(i,:));
    skewness_test(i) = skewness(test_data(i,:));
    kurtosis_test(i) = kurtosis(test_data(i,:));
end



%% Feature extraction Frequency Domain %%
Fs = 2560;          % Sampling frequency
dt = 1/Fs;          % Time step
Ntime = 38400;      % Number of data points
Ttotal = 15;        % Total time
df = 1/Ttotal;      % Fundamental frequency

for i=1:20
    train_healthy_fft(i,:)=fft(train_healthy(i,:));
    train_faulty_ub1_fft(i,:)=fft(train_faulty_ub1(i,:));
    train_faulty_ub2_fft(i,:)=fft(train_faulty_ub2(i,:));

%     figure(i)
%     plot((0:Ntime/2-
1)/Ttotal,(2/Ntime)*abs(train_healthy_fft(i,1:Ntime/2)))
%     xlabel(['$ Frequency \;\mathrm{[Hz]} $'],'interpreter','latex')
%     ylabel(['$ Magnitude \;\mathrm{[V]} $'],'interpreter','latex')
%     txt=['FFT Healthy', num2str(i)];
%     title(txt)
%     grid on
%     xlim([0 50])
%     figure(i+20)
%     plot((0:Ntime/2-
1)/Ttotal,(2/Ntime)*abs(train_faulty_ub1_fft(i,1:Ntime/2)))
%     xlabel(['$ Frequency \;\mathrm{[Hz]} $'],'interpreter','latex')
%     ylabel(['$ Magnitude \;\mathrm{[V]} $'],'interpreter','latex')
%     txt=['FFT Faulty', num2str(i)];
%     title(txt)
%     grid on
%     xlim([0 50])
%     plot((0:Ntime/2-
1)/Ttotal,(2/Ntime)*abs(train_faulty_ub2_fft(i,1:Ntime/2)))
%     xlabel(['$ Frequency \;\mathrm{[Hz]} $'],'interpreter','latex')
%     ylabel(['$ Magnitude \;\mathrm{[V]} $'],'interpreter','latex')
%     txt=['FFT Faulty', num2str(i)];
%     title(txt)
%     grid on
%     xlim([0 50])

    amplitude_healthy_1x(i) =
max(((2/Ntime)*abs(train_healthy_fft(i,1:750/2))));
    amplitude_faulty_ub1_1x(i) =
max(((2/Ntime)*abs(train_faulty_ub1_fft(i,1:750/2))));
    amplitude_faulty_ub2_1x(i) =
max(((2/Ntime)*abs(train_faulty_ub2_fft(i,1:750/2))));
```

```
    amplitude_healthy_2x(i) =
max(((2/Ntime)*abs(train_healthy_fft(i,750/2:750))));
    amplitude_faulty_ub1_2x(i) =
max(((2/Ntime)*abs(train_faulty_ub1_fft(i,750/2:750))));
    amplitude_faulty_ub2_2x(i) =
max(((2/Ntime)*abs(train_faulty_ub2_fft(i,750/2:750))));

end
for i=1:30
    testset(i,:) = fft(test_data(i,:));
    amplitude_testset_1x(i) = max(((2/Ntime)*abs(testset(i,1:750/2))));
    amplitude_testset_2x(i) = max(((2/Ntime)*abs(testset(i,750/2:750))));
end



% figure
% plot(1:20,amplitude_healthy,'-ko')
% hold on
% plot(1:20,amplitude_faulty_ub1,'-r*')
% hold on
% plot(1:20,amplitude_faulty_ub2,'-ro')
% xlabel(['$ Samples\;\mathrm{} $'],'interpreter','latex')
% ylabel(['$ Amplitude\;\mathrm{[V]} $'],'interpreter','latex')
% legend('Healthy Amplitudes','Faulty Amplitudes')
% txt=['Feature extraction'];
% title(txt)
% grid on



%% Feature matrix
%RMS - P2P - Skewness - Kurtosis : order of columns
% feature_mat_healthy = [kurtosis_healthy.' skewness_healthy.' rms_healthy.'
p2p_healthy.' amplitude_healthy_1x.' amplitude_healthy_2x.'];
% feature_mat_faulty_ub1 = [kurtosis_faulty_ub1.' skewness_faulty_ub1.'
rms_faulty_ub1.' p2p_faulty_ub1.' amplitude_faulty_ub1_1x.'
amplitude_faulty_ub1_2x.'];
% feature_mat_faulty_ub2 = [kurtosis_faulty_ub2.' skewness_faulty_ub2.'
rms_faulty_ub2.' p2p_faulty_ub2.' amplitude_faulty_ub2_1x.'
amplitude_faulty_ub2_2x.'];
% feature_mat_test = [kurtosis_test.' skewness_test.' rms_test.' p2p_test.'
amplitude_testset_1x.' amplitude_testset_2x.'];

feature_mat_healthy = [rms_healthy.' p2p_healthy.' amplitude_healthy_1x.'
amplitude_healthy_2x.'];
feature_mat_faulty_ub1 = [rms_faulty_ub1.' p2p_faulty_ub1.'
amplitude_faulty_ub1_1x.' amplitude_faulty_ub1_2x.'];
feature_mat_faulty_ub2 = [rms_faulty_ub2.' p2p_faulty_ub2.'
amplitude_faulty_ub2_1x.' amplitude_faulty_ub2_2x.'];
feature_mat_test = [rms_test.' p2p_test.' amplitude_testset_1x.'
amplitude_testset_2x.'];


D=[feature_mat_healthy; feature_mat_faulty_ub1; feature_mat_faulty_ub2];
D_test=[feature_mat_test];
labels(1:20)={'healthy'};
```

```
labels(21:40)={'UB1'};
labels(41:60)={'UB2'};

%% Data Normalization
D_test=(D_test-mean(D))./(std(D));
D=(D-mean(D))./(std(D));
%% PCA
%%Covariance
C = cov(D)
%%Eigenvalues and Vectors
[evect eval]=eig(C)
eigen_values = diag(eval).'
D_PCA = D*evect
D_test_PCA = D_test*evect

% %%PCA
% [coeff,score]=pca(D)
% D_PCA=D*score

%% plotting Eigen Values
figure();
x_axis = [1:length(eigen_values)];
grid on;
plot(x_axis,eigen_values,'-o');
ylabel('Eigen Values');
title('Eigen values plot for PC selection');

%%SVM

%Mdl = fitcecoc(D_PCA(:,5:6),labels)% if we fit model on D_PCA our test data
is in different format-Ashwin
Mdl = fitcecoc(D(:,3:4),labels)
%test_label=predict(Mdl,D_test_PCA(:,5:6))
test_label=predict(Mdl,D_test(:,3:4))


%% Confusion matrix

true_label(1:10)={'healthy'};
true_label(11:20)={'UB1'};
true_label(21:30)={'UB2'};
true_label_num = grp2idx(true_label); % 1= healthy, 2 = ub1, 3 = ub2
test_label_num = grp2idx(test_label);

confusion_matrix = zeros(3,3);

for i=1:length(true_label)
   confusion_matrix(true_label_num(i),test_label_num(i)) =
confusion_matrix(true_label_num(i),test_label_num(i)) +1;
end
confusion_matrix
hit_rate = trace(confusion_matrix)/sum(sum(confusion_matrix))
```