```
load feature2 %son feature matrisini yükler

%% Veri Seti - Eğitim - Test
L=length(feature2(1,:));

n = length(feature2(:,L));
partitionPart = cvpartition(n,'Holdout',0.3); % Nonstratified partition

index_of_train = training(partitionPart);
table_of_train = feature2(index_of_train,:); % train kısmı toplam veri
index_of_test = test(partitionPart);
table_of_test = feature2(index_of_test,:);  % test kısmı toplam veri

%eğitim kısmı yukarıda train olan kısım
EgitimHyp= table_of_train(:,L);
EgitimFeature= table_of_train;
EgitimFeature(:,L)= [];

TestHyp= table_of_test(:,L);
TestFeature= table_of_test;
TestFeature(:,L)= [];



%% SVM - Sınıflandırıcı
template_svm = templateSVM(...
    'KernelFunction', 'polynomial', ...
    'PolynomialOrder', 2, ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 1, ...
    'Standardize', true);
classificationSVM = fitcecoc( EgitimFeature, EgitimHyp, ...
    'Learners', template_svm, ...
    'Coding', 'onevsone', ...
    'ClassNames', [1; 2; 3; 4; 5]);

Testsonuc_SVM=predict(classificationSVM,TestFeature);
Accuracy_SVM = sum((Testsonuc_SVM == TestHyp))/length(TestHyp)*100;


confusion_matrix_SVM = confusionmat(TestHyp,Testsonuc_SVM);
% confusionchart(TestHyp,Testsonuc_SVM) %confusion matrisini çizdiriyor

% dogrulukToplam=0;
% i=0;
% j=0;
% for i=1:5
%     for j=1:5
%         if(i==j)
%             dogrulukToplam=dogrulukToplam+confusion_matrix_SVM(i,j);
%         end
%     end
% end

%Accuracy2_SVM= dogrulukToplam/length(TestHyp)*100; %yukarıdaki ile aynı
hesaplıyor
```

```matlab
%sensitivity values
sens1 = confusion_matrix_SVM(1,1)/sum(confusion_matrix_SVM(:,1));
sens2 = confusion_matrix_SVM(2,2)/sum(confusion_matrix_SVM(:,2));
sens3 = confusion_matrix_SVM(3,3)/sum(confusion_matrix_SVM(:,3));
sens4 = confusion_matrix_SVM(4,4)/sum(confusion_matrix_SVM(:,4));
sens5 = confusion_matrix_SVM(5,5)/sum(confusion_matrix_SVM(:,5));

%average sensitivity
AvSens_SVM=(sens1+sens2+sens3+sens4+sens5)/5*100;

%specifity values
spec1 = confusion_matrix_SVM(1,1)/sum(confusion_matrix_SVM(1,:));
spec2 = confusion_matrix_SVM(2,2)/sum(confusion_matrix_SVM(2,:));
spec3 = confusion_matrix_SVM(3,3)/sum(confusion_matrix_SVM(3,:));
spec4 = confusion_matrix_SVM(4,4)/sum(confusion_matrix_SVM(4,:));
spec5 = confusion_matrix_SVM(5,5)/sum(confusion_matrix_SVM(5,:));

%average specifity
AvSpec_SVM =(spec1+spec2+spec3+spec4+spec5)/5*100;


%% KNN Classifier

classificationKNN = fitcknn(...
    EgitimFeature, ...
    EgitimHyp, ...
    'Distance', 'Euclidean', ...
    'Exponent', [], ...
    'NumNeighbors', 10, ...
    'DistanceWeight', 'SquaredInverse', ...
    'Standardize', true, ...
    'ClassNames', [1; 2; 3; 4; 5]);

Testsonuc_KNN =predict(classificationKNN,TestFeature);
Accuracy_KNN = sum((Testsonuc_KNN == TestHyp))/length(TestHyp)*100;


confusion_matrix_KNN = confusionmat(TestHyp,Testsonuc_KNN);
% confusionchart(TestHyp,Testsonuc_KNN) %confusion matrisini çizdiriyor

%sensitivity values
sens1_K = confusion_matrix_KNN(1,1)/sum(confusion_matrix_KNN(:,1));
sens2_K = confusion_matrix_KNN(2,2)/sum(confusion_matrix_KNN(:,2));
sens3_K = confusion_matrix_KNN(3,3)/sum(confusion_matrix_KNN(:,3));
sens4_K = confusion_matrix_KNN(4,4)/sum(confusion_matrix_KNN(:,4));
sens5_K = confusion_matrix_KNN(5,5)/sum(confusion_matrix_KNN(:,5));

%average sensitivity
AvSens_KNN=(sens1_K+sens2_K+sens3_K+sens4_K+sens5_K)/5*100;

%specifity values
spec1_K = confusion_matrix_KNN(1,1)/sum(confusion_matrix_KNN(1,:));
spec2_K = confusion_matrix_KNN(2,2)/sum(confusion_matrix_KNN(2,:));
```

```
spec3_K = confusion_matrix_KNN(3,3)/sum(confusion_matrix_KNN(3,:));
spec4_K = confusion_matrix_KNN(4,4)/sum(confusion_matrix_KNN(4,:));
spec5_K = confusion_matrix_KNN(5,5)/sum(confusion_matrix_KNN(5,:));

%average specifity
AvSpec_KNN =(spec1_K+spec2_K+spec3_K+spec4_K+spec5_K)/5*100;

%% DECISION TREE CLASSIFIER

classificationTree = fitctree(...
   EgitimFeature, ...
    EgitimHyp, ...
    'SplitCriterion', 'gdi', ...
    'MaxNumSplits', 20, ...
    'Surrogate', 'off', ...
    'ClassNames', [1; 2; 3; 4; 5]);


Testsonuc_TREE=predict(classificationTree,TestFeature);
Accuracy_TREE = sum((Testsonuc_TREE == TestHyp))/length(TestHyp)*100;


confusion_matrix_TREE = confusionmat(TestHyp,Testsonuc_TREE);
% confusionchart(TestHyp,Testsonuc_KNN) %confusion matrisini çizdiriyor

%sensitivity values
sens1_T = confusion_matrix_TREE(1,1)/sum(confusion_matrix_TREE(:,1));
sens2_T = confusion_matrix_TREE(2,2)/sum(confusion_matrix_TREE(:,2));
sens3_T = confusion_matrix_TREE(3,3)/sum(confusion_matrix_TREE(:,3));
sens4_T = confusion_matrix_TREE(4,4)/sum(confusion_matrix_TREE(:,4));
sens5_T = confusion_matrix_TREE(5,5)/sum(confusion_matrix_TREE(:,5));

%average sensitivity
AvSens_TREE=(sens1_T+sens2_T+sens3_T+sens4_T+sens5_T)/5*100;

%specifity values
spec1_T = confusion_matrix_TREE(1,1)/sum(confusion_matrix_TREE(1,:));
spec2_T = confusion_matrix_TREE(2,2)/sum(confusion_matrix_TREE(2,:));
spec3_T = confusion_matrix_TREE(3,3)/sum(confusion_matrix_TREE(3,:));
spec4_T = confusion_matrix_TREE(4,4)/sum(confusion_matrix_TREE(4,:));
spec5_T = confusion_matrix_TREE(5,5)/sum(confusion_matrix_TREE(5,:));

%average specifity
AvSpec_TREE =(spec1_T+spec2_T+spec3_T+spec4_T+spec5_T)/5*100;


SONUC =[Accuracy_SVM   AvSens_SVM  AvSpec_SVM  ;
       Accuracy_KNN   AvSens_KNN  AvSpec_KNN  ;
       Accuracy_TREE  AvSpec_KNN  AvSpec_TREE];

save('SONUC.mat','SONUC');
```

```
AvAcc_AvSens_AvSpec=(SONUC1+SONUC2+SONUC3+SONUC4+SONUC5+SONUC7+SONUC8+SONUC11+
SONUC12+SONUC20)/10;

save('SONUC_EN_SON.mat','AvAcc_AvSens_AvSpec');


%% ARTIFICIAL NEURAL NETWORK

Feature=feature20;  %feature20 kısmı girdiye göre kalanı aynı
featureHYP=Feature(:,21);
Feature(:,21)=[];
Feature=Feature'; %ANN toolboxına transposeu alınmış şeklini giriyoruz.

lengthFeature=length(Feature);

targetMat=zeros(5,lengthFeature); % 5=class sayısı, bizde hep 5, stagelere
göre olduğundan

for i=1:lengthFeature

if(featureHYP(i)==1)
    targetMat(1,i)=1;
end


if(featureHYP(i)==2)
    targetMat(2,i)=1;
end


if(featureHYP(i)==3)
    targetMat(3,i)=1;
end


if(featureHYP(i)==4)
    targetMat(4,i)=1;
end


if(featureHYP(i)==5)
    targetMat(5,i)=1;
end

end

% ANN'ye input olarak Feature matrisini, target olarak da targetMat matrisini
veriyoruz
```

PREPROCESSING ARTI FEATURE EXTRACTION KISMI

```
%SLEEP STAGE NUMBER HYPNOGRAM
%Kisi 1
dataHyp = edfread('SC4001EC-Hypnogram.edf');
infoHyp = edfinfo('SC4001EC-Hypnogram.edf');

dataPSG = edfread('SC4001E0-PSG.edf');
infoPSG = edfinfo('SC4001E0-PSG.edf');

%Kisi 2
dataHyp = edfread('SC4002EC-Hypnogram.edf');
infoHyp = edfinfo('SC4002EC-Hypnogram.edf');

dataPSG = edfread('SC4002E0-PSG.edf');
infoPSG = edfinfo('SC4002E0-PSG.edf');

%Kisi 3
dataHyp = edfread('SC4011EH-Hypnogram.edf');
infoHyp = edfinfo('SC4011EH-Hypnogram.edf');

dataPSG = edfread('SC4011E0-PSG.edf');
infoPSG = edfinfo('SC4011E0-PSG.edf');

%Kisi 4
dataHyp = edfread('SC4012EC-Hypnogram.edf');
infoHyp = edfinfo('SC4012EC-Hypnogram.edf');

dataPSG = edfread('SC4012E0-PSG.edf');
infoPSG = edfinfo('SC4012E0-PSG.edf');

%Kisi 5
dataHyp = edfread('SC4021EH-Hypnogram.edf');
infoHyp = edfinfo('SC4021EH-Hypnogram.edf');

dataPSG = edfread('SC4021E0-PSG.edf');
infoPSG = edfinfo('SC4021E0-PSG.edf');

%Kisi 6
dataHyp = edfread('SC4022EJ-Hypnogram.edf');
infoHyp = edfinfo('SC4022EJ-Hypnogram.edf');

dataPSG = edfread('SC4022E0-PSG.edf');
infoPSG = edfinfo('SC4022E0-PSG.edf');

%Kisi 7
dataHyp = edfread('SC4031EC-Hypnogram.edf');
infoHyp = edfinfo('SC4031EC-Hypnogram.edf');

dataPSG = edfread('SC4031E0-PSG.edf');
infoPSG = edfinfo('SC4031E0-PSG.edf');

%Kisi 8
```

```
dataHyp = edfread('SC4032EP-Hypnogram.edf');
infoHyp = edfinfo('SC4032EP-Hypnogram.edf');

dataPSG = edfread('SC4032E0-PSG.edf');
infoPSG = edfinfo('SC4032E0-PSG.edf');

%Kisi 9
dataHyp = edfread('SC4041EC-Hypnogram.edf');
infoHyp = edfinfo('SC4041EC-Hypnogram.edf');

dataPSG = edfread('SC4041E0-PSG.edf');
infoPSG = edfinfo('SC4041E0-PSG.edf');

%Kisi 10
dataHyp = edfread('SC4042EC-Hypnogram.edf');
infoHyp = edfinfo('SC4042EC-Hypnogram.edf');

dataPSG = edfread('SC4042E0-PSG.edf');
infoPSG = edfinfo('SC4042E0-PSG.edf');

%%
infoPSG.SignalLabels
fs = infoPSG.NumSamples/seconds(infoPSG.DataRecordDuration);

signum = 2; %sütunda ilerliyor EEG EOG vb.  %Parietal elektrot (2) ile
çalışıyoruz
%Frontal Elektrot=1
t = (0:infoPSG.NumSamples(signum)-1)/fs(signum);


lengtHYP= length(infoHyp.Annotations.(2));

for i=2:1:(lengtHYP-2)
    s(i-1,1)=infoHyp.Annotations.(2)(i);  % s matrisi: 120sec vb.
end

x=seconds(s); % x matrisi: 120 çekiyor
a=x/30;  % a matrisi: 120/30=4 kaç epoch o değerden olduğunu söylüyor

for i=2:1:(lengtHYP-2)
    k(i-1,1)=infoHyp.Annotations.(1)(i); % k matrisi: 'Sleep Stage 1' vb.
string matrisi
end

wake='Sleep stage W';
stage1='Sleep stage 1';
stage2='Sleep stage 2';
stage3='Sleep stage 3';
stage4='Sleep stage 4';
rem='Sleep stage R';
movTime='Movement time';


for i=1:1:(lengtHYP-3)
```

```
    if strcmp(k(i),stage1)
       f(i,1)=1;                         % f matrisi: sırasıyla 5 1 2 vb (kaç 
epoch old. bağımsız)
    end

       if strcmp(k(i),stage2)
          f(i,1)=2;
       end

         if strcmp(k(i),stage3) | strcmp(k(i),stage4)
            f(i,1)=3;
         end

            if strcmp(k(i),rem)
               f(i,1)=4;
            end

               if strcmp(k(i),wake)
                  f(i,1)=5;
               end

                   if strcmp(k(i),movTime)
                      f(i,1)=0;   %movement time'ı en sonda atacağız
                   end

end


o=1;
for i=1:1:(lengtHYP-3)
    for j=1:1:a(i)

          feature(o,1)=f(i);  % feature matrisine 1021 tane 5 yazdırma kısmı
           o= o+1;

    end
end


%ÖZELLİK BÖLÜMÜ

firstWake= seconds(infoHyp.Annotations.(2)(1))/30;
lastWake= seconds(sum(infoHyp.Annotations.(2))-
infoHyp.Annotations.(2)(lengtHYP-1)-infoHyp.Annotations.(2)(lengtHYP))/30;

for j=firstWake+1:1:lastWake
    y = dataPSG.(signum){j};
    recnum = j-firstWake;  %satırda ilerliyor record 2 3 vb.

   %PREPROCESSING
   %1)NORMALISATION
   N = normalize(y);
   %2)OUTLIER
   thr=(2/4)*max(N);
for i=1:length(N)
    if N(i)>thr
        y1(i)=mean(N);
    else if N(i)<-thr
            y1(i)=-mean(N);
        else
            y1(i)=N(i);
        end
    end
end

%3)BANDPASS
b  = bandpass(y1,[0.5 13],100);  %alfa aldık, accuracy ye göre wakeEkstrayı
ekleyip 35]e çeviririz (betaya)


% FEATURE EXTRACTION
%1)ZERO CROSSING       %doğru çalışıyor kontrol ettik.
zc=0;

for i=1:length(b)-1
    if b(i)*b(i+1)<0
        zc=zc+1;
    end
end
feature(recnum,2)=zc;%feature matrisinde ilk sütünda kayıt yapıyor.


%2) PETROSIAN FRACTAL DIMENSION

for k=1:length(b)
    if b(k)>mean(b)
        y2(k)=1;
    else b(k)<=mean(b);
            y2(k)=-1;
    end
end

y2=transpose(y2);

zcp=0;

for i=1:length(y2)-1
    if y2(i)*y2(i+1)<0
        zcp=zcp+1;
    end
end

L=length(y2);
%ZC2=number of sign changes

P=log10(L)/(log10(L)+log10(L/(L+0.4*zcp)));
```

```
feature(recnum,3)=P;

%3) MAX MIN DISTANCE (MMD)

for i=1:1:length(b)
    if b(i)==max(b)
        xmax=i;
    else if b(i)==min(b)
            xmin=i;
        end
    end
end

MMD=sqrt(((max(b)-min(b)).^2)+((xmax-xmin).^2));
feature(recnum,4)=MMD;


%4-5) HJORT PARAMETERS
Mobility=sqrt(var(diff(b))/var(b));

Complexity=sqrt(var(diff(diff(b)))/var(b))/sqrt(var(diff(b))/var(b));

feature(recnum,5)=Mobility;
feature(recnum,6)=Complexity;


%6) VARIANCE
 varyans=var(b);
 feature(recnum,7)= varyans;

%7)MEDIAN FREQUENCY
 medianFreq=medfreq(b);
 feature(recnum,8)= medianFreq;


%8-9-10) BANDPOWER (for 3 frequency band)
 pbandDelta = bandpower(b,100,[0.4 4]);
 pbandTheta = bandpower(b,100,[4 8]);
 pbandAlfa = bandpower(b,100,[8 13]);

 feature(recnum,9)= pbandDelta;
 feature(recnum,10) = pbandTheta;
 feature(recnum,11) = pbandAlfa;


% DISCRETE WAVELET TRANSFORM
 waveletName='db5';
 level=5;
% 1-D wavelet decomposition
 [c0,l0]=wavedec(b,level,waveletName);
% 1-D detail coefficients
 %cD3 = detcoef(c0,l0,3); %BETA
 cD4 = detcoef(c0,l0,4); %ALPHA
 cD5 = detcoef(c0,l0,5); %THETA
 cA5 = appcoef(c0,l0,waveletName,5); %DELTA

% Reconstruct single branch from 1-D wavelet coefficients
 %D3 = wrcoef('d',c0,l0,waveletName,3); %BETA
 D4 = wrcoef('d',c0,l0,waveletName,4); %ALPHA
 D5 = wrcoef('d',c0,l0,waveletName,5); %THETA
 A5 = wrcoef('a',c0,l0,waveletName,5); %DELTA
 %plot(t,D4,'r',t,D5,'g',t,A5,'y')  %Dalga boyu artıkça  frekans azalır

 %11-12-13) APPROXIMATE ENTROPY
 appEntDelta= approximateEntropy(A5);
 appEntTheta= approximateEntropy(D5);
 appEntAlfa= approximateEntropy(D4);

 feature(recnum,12) = appEntDelta;
 feature(recnum,13) = appEntTheta;
 feature(recnum,14) = appEntAlfa;


% FEATURES APPLIED ON THE FIRST SIGNAL

thr=(2/4)*max(y);
for i=1:length(y)
    if y(i)>thr
        new(i)=mean(y);
    else if y(i)<-thr
            new(i)=-mean(y);
        else
            new(i)=y(i);
        end
    end
end

ilksinyal=bandpass(new,[0.4 13],50);

% 14) KURTOSIS
 kur = kurtosis(ilksinyal);
 feature(recnum,15)= kur;

% 15) SKEWNESS
skew = skewness(ilksinyal);
feature(recnum,16)= skew;

% 16) MEAN
feature(recnum,17)= mean(ilksinyal);

% 17) STANDART DEVIATION
feature(recnum,18)= std(ilksinyal);

 %18)RMS
 feature(recnum,19) = rms(ilksinyal);

 %19)MEAN ABSOLUTE DEVIATION
 feature(recnum,20)= mad(ilksinyal);

 %20)INTERQUATRILE RANGE
 feature(recnum,21) = iqr(ilksinyal);

end


ozellikSayisi=20;  % en son feature sayısına göre değiştiririz

feature(:,ozellikSayisi+2)=feature(:,1);  %en son boş satıra ilk sütunu
atayacak
feature(:,1)=[]; %ilk sütunu silecek


for i=1:length(feature)
    if feature(i,21)==0
        feature(i,:)=[];
    end
end


%% bu kısım kişi numarasına göre değişecek
feature12=feature;

save('feature12.mat','feature12');
% feature matrisi son hali: sütunlar featurelar, son sütun 1 2 3 4 5ler
```