ELM 472
DRONE SINIFLANDIRMA PROJESİ

KAMİL ALP ÇINAR
<k.cinar2018@gtu.edu.tr> Elektronik Mühendisliği Bölümü, GTÜ, İstanbul, Türkiye

ÖZET:

I. GİRİŞ

Birbirinden farklı sınıflara(5) ait drone seslerinden makine öğrenmesi/derin öğrenme algoritmalarıyla sınıflandırma yapmak ve sonuçları birbirleriyle karşılaştırarak en iyi modeli/yöntemi bulmak.

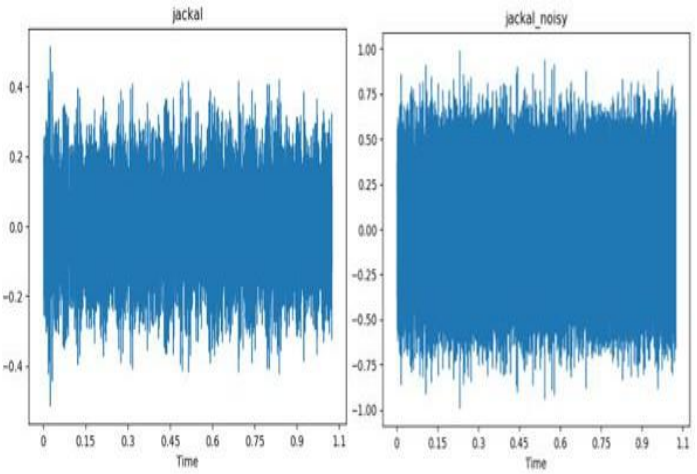
II. VERİSETİ TOPLAMA

Verisetinde 5 farklı drone sesi var:Bebop,Membo,Savunma Enstitüsündeki drone,Jackal(FlyBvlos verdi.),6 pervaneli drone(Youtube).2 tanesini(bebop,membo) internetten indirdik.Savunma Enstitüsündeki drone ‘u uçurup sesini kaydettik.Jackal için flybvlos ‘a gidip yetkili biriyle konuştuk ve sesi aldık.6 pervaneli drone sesini Youtubeda bir videodan aldık.

III. VERİSETİ DÜZENLEME

Drone pervane sesi devamlı aynı hareketi yaptığı için sesi hem fazlalaştırmak için hemde daha efektif kullanabilmek adına 1,1 sn’lik parçalara ayırdık. Seslerimizi çoğaltmak adına gürültü,yankı ve ses kayırma gibi metodlar uyguladık.

Şekil 1. Seslerin 1,1 saniyelik görünümü



Şekil 2. Jackal sesinin normal hali ve gürültü eklenmiş hali

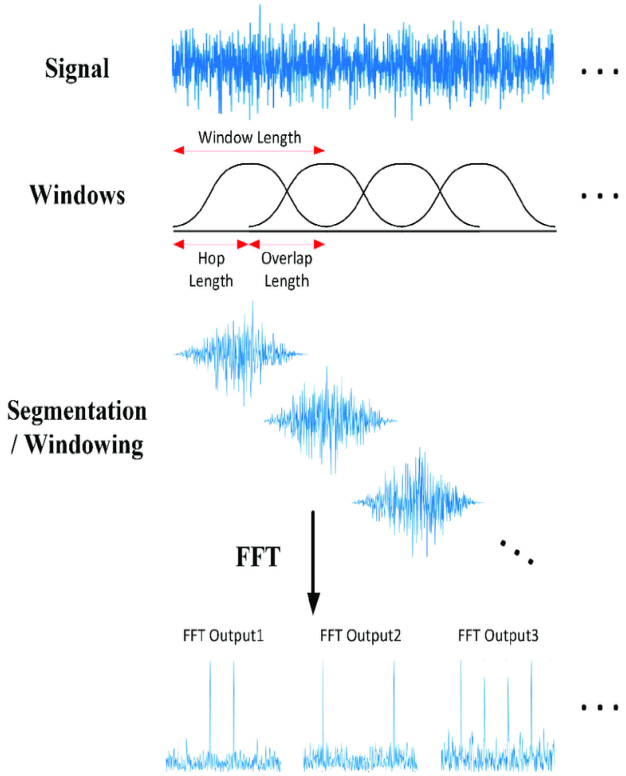
STFT: Short Time Fourier Transform(STFT): sürekli bir sinyali zaman ve frekans alanlarında incelemek için kullanılabilen bir yöntemdir.Bir sinyali belirli aralıklarda parçalara ayırarak, her parçanın Fourier dönüşümünü alarak sinyali zamanda ve frekansda inceleyebilmemizi sağlar. STFT, sinyalin frekans bileşenlerinin zamandaki değişimini inceleyebilmek için kullanılır. STFT alınırken, öncelikle bir pencere fonksiyonu seçilir. Pencere fonksiyonu(W), sinyali zamanda parçalara ayırırken kullanılan matematiksel bir fonksiyondur ve sinyalin frekans bileşenlerinin bozulmaması için dikkatli bir şekilde seçilmesi gerekir. Daha sonra, sinyal parçalara ayrılır ve her parçanın Fourier dönüşümü alınır. Bu işlem, sinyalin zamandaki değişimini inceleyebilmemizi sağlar.

$$STFT = X(\tau,\omega) = \int_{-\infty}^{\infty} x(t)w(t-\tau)e^{-i\omega t}dt$$

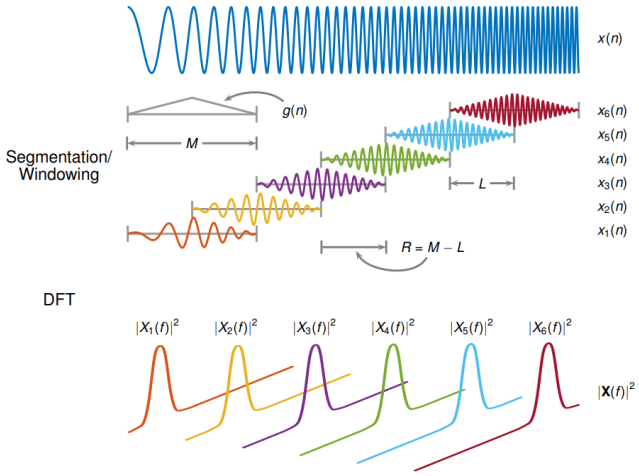
Şekil3. Sürekli STFT Formülü

$$STFT = X(m,\omega) = \sum_{n=-\infty}^{\infty} x_n w_{n-m} e^{-i\omega t_n}$$

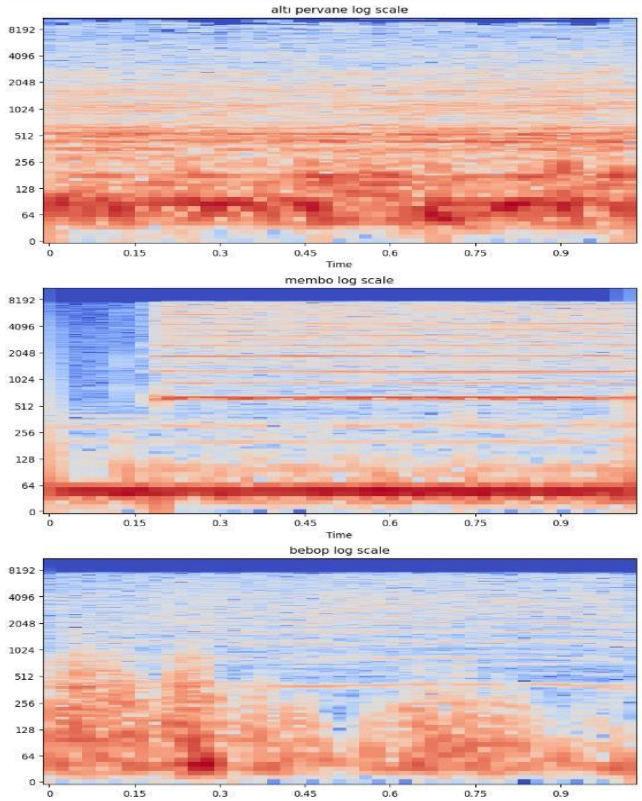
Şekil4. Ayrık STFT Formülü



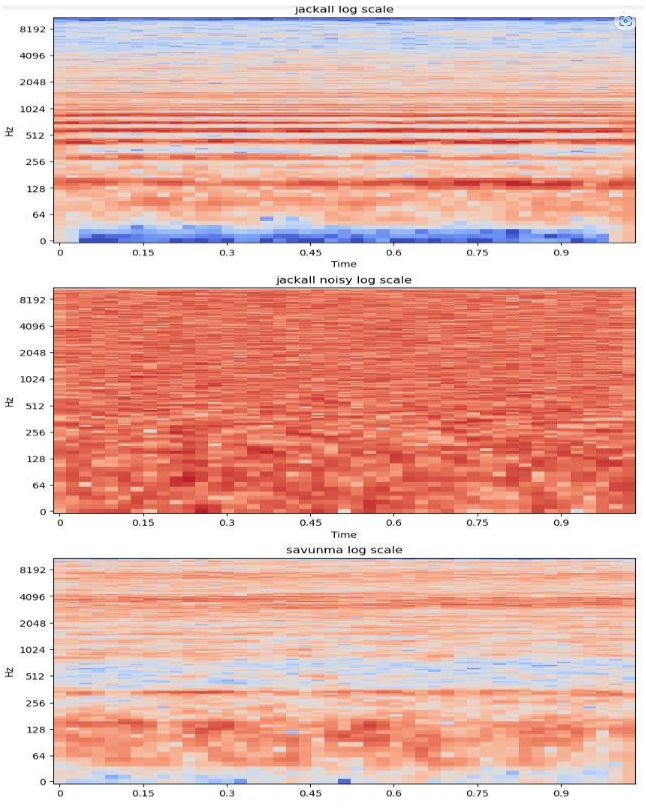
Şekil5. STFT diagramı **[1]**



Şekil5. STFT diagramı



Şekil6. Seslerin STFT alınarak logaritmik eksende spektrogramları



Şekil6. Seslerin STFT alınarak logaritmik eksende spektrogramları

```
from pydub import AudioSegment
alti_pervane="alti_perva.wav"

# Read the input file as a NumPy array
data,sr = librosa.load(alti_pervane)

# Generate white noise using Scipy
noise = scipy.randn(len(data))

# Add the noise to the original data
noisy_data = data + noise

# Convert the noisy data back into a sound file
segment = AudioSegment(
    noisy_data.tobytes(),
    frame_rate=sample_rate,
    sample_width=sample_width,
    channels=num_channels
)

# Save the noisy sound file
segment.export("noisy_alti.wav", format="wav")
```

Şekil7.gürültü ekleme kod bloğu

```
from pydub import AudioSegment


audio = AudioSegment.from_file("alti_perva.wav", format="wav")

# Add reverb to the audio
reverbed = audio.apply_gain(10).low_pass_filter(500).high_pass_filter(500).low_pass_filter(1000).high_

# Save the reverb-added audio
reverbed.export("alti_reverbed.wav", format="wav")
```

Şekil8. Yankı ekleme kod bloğu

```
data, sample_rate = sf.read("alti_perva.wav")

# Shift the pitch up by one semitone (100 cents)
data = data * 2**(100/1200)

sf.write("alti_shifted.wav", data, sample_rate)
```

Şekil9.Ses kaydırma kod bloğu

Sesler bu şekilde çoğaltıldıktan her birini librosa kütüphanesinin bize sunmuş olduğu librosa.load() komutu ile okuduk ve sayılara çevirdik.Librosa kütüphanesi sesleri normalize ederek sayıya dönüştürebiliyor ve sample rate her zaman 22050 alınıyor.

Daha sonra etiketlemek için panda framework kullandık.Bütün sınıfların kendi panda frameworklerine geçirip etiketledik.

df_savunma

	Name	Data	Sample Rate	Length	label
0	drone_savunma.wav	[-0.033083413, -0.09698918, -0.009048275, 0.07...	22050	40.446032	2.0
1	drone_savunma0.wav	[-0.033083413, -0.09698918, -0.009048275, 0.07...	22050	1.024036	2.0
2	drone_savunma1.wav	[0.029927712, -0.013015142, -0.061501004, -0.0...	22050	1.024036	2.0
3	drone_savunma10.wav	[-0.025829073, -0.019390129, -0.0019575886, 0....	22050	1.024036	2.0
4	drone_savunma11.wav	[-0.00040498, 0.0010001346, 0.0010421382, 0.00...	22050	1.024036	2.0
...	...	...	...	...	...
159	savunma_shifted5.wav	[-0.02156402, 0.010027022, -0.012186344, -0.02...	22050	1.024036	2.0
160	savunma_shifted6.wav	[0.0060531446, -0.015906442, -0.056181155, -0....	22050	1.024036	2.0
161	savunma_shifted7.wav	[-0.037787493, -0.033458345, 0.04639002, 0.011...	22050	1.024036	2.0
162	savunma_shifted8.wav	[0.031056762, 0.032730732, -0.011158691, -0.04...	22050	1.024036	2.0
163	savunma_shifted9.wav	[0.0011564334, 0.02844385, 0.038972825, 0.0377...	22050	1.024036	2.0

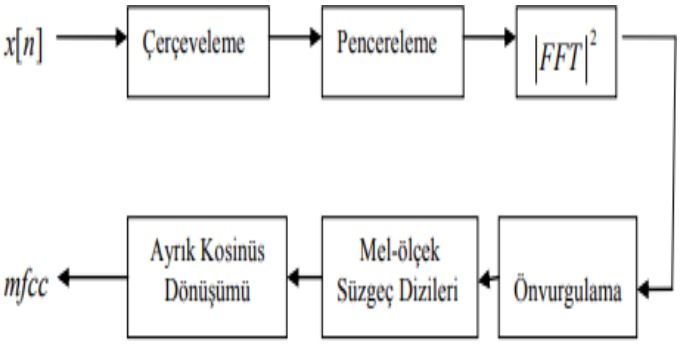
164 rows × 5 columns

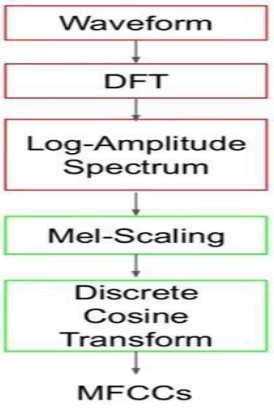
Şekil10. Savunma isimli drone 'un pandaframework 'ü

IV. ÖZNITELIK ÇIKARIMI

Öznitelik olarak MFCC(MEL FREQUENCY CEPSTRAL COEFFİCENTS) ' ı seçtik.

MFCCs:

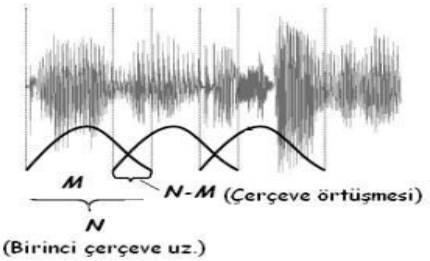




Şekil11. MFCC özniteliklerinin çıkarılma işleminin blok diyagramı

A. Çerçeveleme:

Bir seste pervane bir kaç kez döndüğü için tüm işaretin FFT'sini almak yerine çerçevenin FFT'si hesaplanır. Çerçeve uzunluğu 10-30 msn arasında değişir. Çerçevelerin örtüşme oranı, çerçeve uzunluğunun % 30'u ile % 75' i arasında alınır (Kinnunen, 2003). Örtüşme uygulanması ile çerçeve sonundaki işaretin önemlerini kaybetmemesi sağlanır.Ses işareti N örnek uzunluğunda konuşma parçalarına bölünür. İlk çerçeve N örnekten oluşurken sonraki çerçeve ilk çerçeveden M örnek sonra başlar ve böylece N-M örnek örtüşür (Rabiner ve Juang, 1993). **Şekil 12** 'de bir konuşma işareti üzerinde çerçeveleme işlemi görülmektedir. **[2]**



şekil12.

B. Pencereleme:
Mel frekansı kepstrum katsayılarını elde etmek için ikinci yapılan işlem pencerelemedir. Pencerelemenin amacı çerçeveleme işlemi sonucunda oluşan spektral etkilerin azaltılmasıdır. Pencereleme ile çerçevelerde süreksizliğin önüne geçilir (Rabiner ve Juang, 1993). Bu sayede sesin orta bölgeleri güçlendirilirken kenar bölgeleri zayıflatılır.
Yaygın olarak kullanılan Hamming, Hanning, Blackman, Gauss, dikdörtgen pencereleme fonksiyonlarının matematiksel ifadeleri aşağıdaki gibidir. **[2]**

Hamming:

$$w[k+1] = 0.54 - 0.46\cos\left(2\pi\frac{k}{N-1}\right) \qquad k = 0,.....$$

ŞEKİL 13.

Hanning:

$$w[k+1] = 0.5\left(1 - \cos\left(2\pi\frac{k}{N-1}\right)\right), \qquad k = 0,.....$$

ŞEKİL 14.

Blackman:

$$w[k+1] = 0.42 - 0.5\cos\left(2\pi\frac{k}{N-1}\right) + 0.08\cos\left(4\pi\frac{k}{N-}\right.$$

ŞEKİL 15.

Gauss:

$$w[k+1] = e^{-\frac{1}{2}\left(\alpha\frac{k-\frac{N}{2}}{N/2}\right)^2} \qquad 0 \le k \le N \qquad \text{ve} \qquad \alpha \ge 2$$

ŞEKİL 16.

Dikdörtgen:

$$w[k+1] = 1, \qquad k = 0,....., N-1$$

ŞEKİL 17.

C. Hızlı Fourier Dönüşümü(FFT):
MFCC elde edilmesinde, pencereden geçirilen işaretin genlik spektrumu FFT ile hesaplanır. FFT ile N örnekten oluşan zaman alanındaki her bir çerçeve, frekans alanına çevrilir.

$$X[k] = \sum_{n=0}^{N-1} x_n \cdot e^{-2\pi jkn/N}, \qquad k = 0,1,2,..., N-1$$

ŞEKİL 18. ÇERVENİN AYRIK FOURİER DÖNÜŞÜMÜ

FFT sonucu kompleks sayıdır. X[k] ;k herhangi bir sayı olsun sonucu real+imajiner olarak ifade edilir.
Buradan

$$|X[k]| = \sqrt{X_{re}[k]^2 + X_{im}[k]^2}$$

Şekil19. k. harmonik bileşene ait genlik

$$\angle X[k] = \tan^{-1}\left(\frac{X_{im}[k]}{X_{re}[k]}\right)$$

Şekil20. k. harmonik bileşene ait faz

Bir işaretin FFT'si hesaplanırken işaretin uzunluğu 2'nin kuvvetleri şeklinde olmalıdır. Örneğin işaret 400 örnekten oluşuyorsa işaretin uzunluğu 512 olana kadar işarete sıfır eklenir ve bu şekilde FFT'si hesaplanır. İşaretin başına veya sonuna sıfır eklenmesi FFT sonucunu değiştirmez.Bunun sebebi log2 tabanıyla alakalıdır.

D. : Logaritmik genlik spektrum:
Bu aşama logaritmik spektrum elde etmek için yapılır.

E: Mel skalasına dönüştürme:
F Hertz'i m Mel'e çevirmek için yaygın bir formül, O'Shaughnessy (1987)'in 10luk tabandaki 2595'li formulüdür. 1127 katsayısıyla kullanılan doğal logaritmalı formül günümüzde daha çok kullanılır. Daha eski yayınlar genellikle 700 Hz yerine 1000 Hz'lik kırılma frekansını kulanır.[vikipedi]

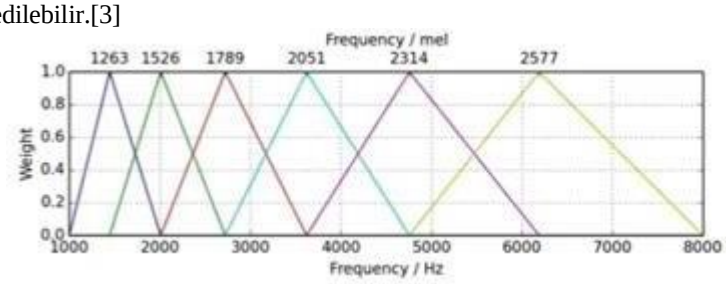
$$m = 2595\log_{10}\left(\frac{f}{700}+1\right) = 1127\log_e\left(\frac{f}{700}+1\right)$$

Şekil21.

$$f = 700(10^{m/2595} - 1) = 700(e^{m/1127} - 1)$$

Şekil22.

Mel filter bank, ses sinyallerinin frekans bileşenlerinin Mel skalasına çevrilmesi için kulllılan bir ölçektir.Bu sayede ses sinyallerinin frekans bileşenleri daha anlamlı bir ölçekte ölçülebilir.Mel filter bank bir ses sinyali üzerinde bir dizi filtre uygular.Bu filtreler ,belirli frekans aralıklarını geçirirler ve diğer frekans aralıklarını engellerler.Bu sayede , ses sinyali üzerinde belirli frekans aralıklarını vurgulayarak, ses sinyali hakkında daha anlamlı bilgiler elde edilebilir.[3]



Şekil23. Mel Filter Bank[4]

ELM472 Makine Öğrenmesinin Temelleri

F: Ayrık Kosinüs Dönüşüm:
Kepstral katsayılar, yüksek derecede istatiksel bağımsızlık gösterip genlik spektrum gösteriminden daha yüksek tanıma oranı verirler. Gerçel kepstrum, logaritmik genlik spektrumun ters fourier dönüşümü olarak tanımlanıp, gerçel işaretler için kosinüs dönüşümü kullanılarak hesaplanır. Mel frekansı kepstrum katsayıları, MFCC(i), süzgeç çıkışlarından aşağıdaki
gibi hesaplanır.

$$MFCC(i) = \frac{1}{FS}\sum_{l=1}^{FS} mfb(l)\cos\left(i\left(l-\frac{1}{2}\right)\frac{\pi}{FS}\right), \qquad i$$

Şekil24.
Ek bilgi:
Neden kosiniüs ,sinus değilin cevabı:
Çünkü cos(0) =1 eğer sin(0)=0 olarak başlarsa bir şeyler kaçabilir o yüzden cosinüs.[5]
Bütün bu adımları librosanın .feature kütüphanesindeki fonksiyon ile hesapladık.

librosa.feature.mfcc

librosa.feature.mfcc(*, y=None, sr=22050, S=None, n_mfcc=20, dct_type=2, norm='ortho', lifter=0, **kwargs) [source]

Mel-frequency cepstral coefficients (MFCCs)

Warning

If multi-channel audio input y is provided, the MFCC calculation will depend on the peak loudness (in decibels) across all channels. The result may differ from independent MFCC calculation of each channel.

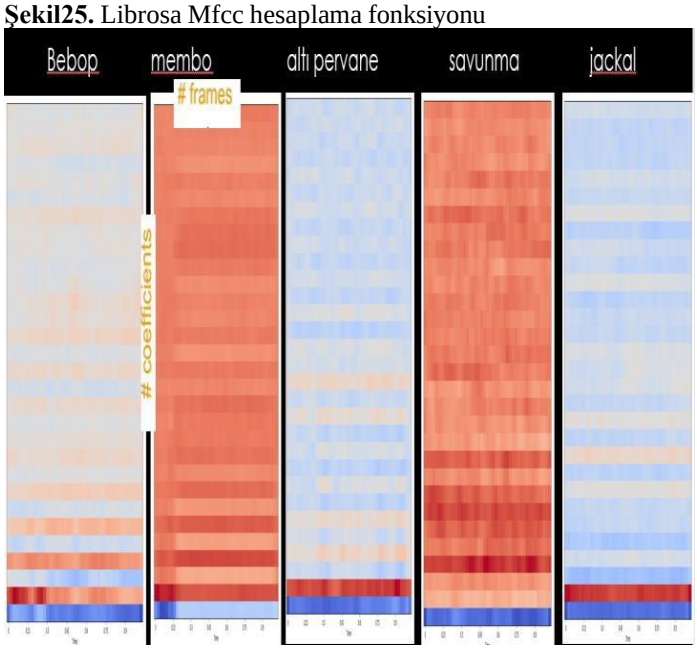
y : np.ndarray [shape=(..., n,)] or None
audio time series. Multi-channel is supported..

sr : number > 0 [scalar]
sampling rate of y

S : np.ndarray [shape=(..., d, t)] or None
log-power Mel spectrogram

n_mfcc : int > 0 [scalar]
number of MFCCs to return

Şekil25. Librosa Mfcc hesaplama fonksiyonu



Şekil 26. Seslerin Mfccs örnekleri

Sınıflandırma işlemimizi **şekil26**'da gözüken MFCCs spektrogramları üzerinden gerçekleştirirken birinci katsayının bütün framelerdeki değerlerinin ortalamasını alıp 1.indekse ; ikinci katsayının bütün framelerdeki değerlerinin ortalamasını alıp 2.indekse ; bu şekilde 30 indeks olacak şekilde 30 luk bir tek boyutlu np.array elde ettik ve sınıflandırmamızı bunun üzerinden yaptık.Aşağıdaki görselde shape leri üzerinden daha net anlaşılacaktır.

```
In [119]: mfccs_features_denemes.shape

Out[119]: (30, 8371)

In [124]: np.mean(mfccs_features_denemes[0])

Out[124]: -216.98279

In [127]: np.mean(mfccs_features_denemes[1])

Out[127]: 102.327896

In [125]: mfccs_scaled_feature__denemes

Out[125]: array([-216.98279   ,  102.327896  ,    3.831232  ,    3.4902709 ,
                   -3.142393  ,   10.741303  ,    1.919243  ,    2.4653356 ,
                  -14.025877  ,    3.5660298 ,  -11.461523  ,   -5.3155904 ,
                   -1.918632  ,   -6.557942  ,    0.2965326 ,   -0.37583044,
                   -3.8639274 ,   -6.016836  ,   -4.025874  ,   -2.129945  ,
                   -7.7792387 ,   -3.3351161 ,   -5.2639823 ,   -1.1952757 ,
                   -5.5687637 ,   -0.73579913,   -6.209263  ,   -2.8340197 ,
                   -4.3865156 ,   -2.7526286 ], dtype=float32)

In [128]: mfccs_scaled_feature__denemes.shape

Out[128]: (30,)
```

Şekil 27. MFCCs 'in 30 a indirgenmiş hali.(1 ses verisi için)

Bu şekilde bütün sadece etiket ve MFCCs leri olan bir panda framework elde etmiş olduk.

```
sade_df
```

	label	feature
474	4	[-148.22943, 68.812645, 48.022465, -5.9262795,...
127	4	[47.729355, -4.7016134, 0.09812409, 9.17263, 4...
248	4	[-226.84262, 182.08871, -69.25542, -9.429601, ...
142	4	[74.04281, 2.0522408, -1.3516281, 1.396056, -3...
354	4	[-36.58694, 90.487755, -21.966702, 11.880893, ...
...	...	...
57	0	[-306.02847, 202.53732, -54.138206, 1.4032874,...
58	0	[-284.5141, 198.9869, -44.732407, 1.5676112, -...
21	0	[-120.90532, 123.900475, 4.218556, 18.608982, ...
32	0	[70.5716, -0.19014108, -2.781077, 3.9726536, -...
94	0	[-99.904945, 114.54688, -0.45474747, 22.29809,...

2066 rows × 2 columns

şekil 28. Etiket ve MFCCs panda framework

ELM472 Makine Öğrenmesinin Temelleri

V.)KULLANILAN SINIFLANDIRMA METODLARI,METRİKLER VE HİPERPARAMETRELER:

Toplam 8 farklı deneme yaptık. 3 tanesi makine öğrenmesi algoritması 5 tanesi deirn öğrenmes

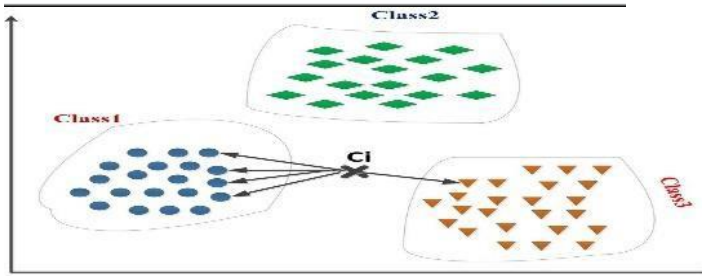
1) K-Nearest Neighbours(KNN)
2) Support Vector Machine(SVM)
3) Decision Tree
4) Derin öğrenme (5 farklı hiperparametre değişikliği)

1) K-Nearest Neighbours:
Algoritmanın çalışmasında bir K değeri belirlenir. Bu K değerinin anlamı bakılacak eleman sayısıdır. Bir değer geldiğinde en yakın K kadar eleman alınarak gelen değer arasındaki uzaklık hesaplanır. Uzaklık hesaplama işleminde genelde Öklid fonksiyonu kullanılır. Öklid fonksiyonuna alternatif olarak Manhattan, Minkowski ve Hamming fonksiyonlarıda kullanılabilir. Uzaklık hesaplandıktan sonra sıralanır ve gelen değer uygun olan sınıfa atanır.[6]

Biz öklid kullandık.

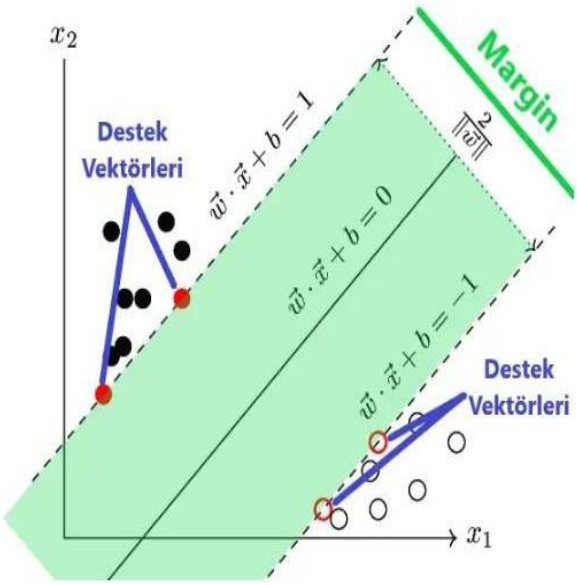
Öklid formülü:

$$\text{Euclidean distance (d)} = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$$



Şekil 29.öklid formülü
şekil30. KNN örnek diagram

2) SUPPORT VECTOR MACHİNES:
Destek Vektör Makineleri (Support Vector Machine) genellikle sınıflandırma problemlerinde kullanılan gözetimli öğrenme yöntemlerinden biridir. Bir düzlem üzerine yerleştirilmiş noktaları ayırmak için bir doğru çizer. Bu doğrunun, iki sınıfının noktaları için de maksimum uzaklıkta olmasını amaçlar. Karmaşık ama küçük ve orta ölçekteki veri setleri için uygundur.[7]



Şekil 31.
Tabloda siyahlar ve beyazlar olmak üzere iki farklı sınıf var. Sınıflandırma problemlerindeki asıl amacımız gelecek verinin hangi sınıfta yer alacağını karar vermektir. Bu sınıflandırmayı yapabilmek için iki sınıfı ayıran bir doğru çizilir ve bu doğrunun ±1'i arasında kalan yeşil bölgeye Margin adı verilir. Margin ne kadar geniş ise iki veya daha fazla sınıf o kadar iyi ayrıştırılır. .[7]

$$\hat{y} = \begin{cases} 0 \text{ if } \mathbf{w}^T \cdot \mathbf{x} + b < 0, \\ 1 \text{ if } \mathbf{w}^T \cdot \mathbf{x} + b \geq 0 \end{cases}$$

Şekil 32.
Aslında değişen pek bir şey yok. w; ağırlık vektörü (θ1), x; girdi vektörü, b; sapmadır (θ0). Yeni bir değer için çıkan sonuç 0'dan küçükse, beyaz noktalara daha yakın olacaktır. Tam tersi, çıkan sonuç 0'a eşit veya büyükse, bu durumda siyah noktalara daha yakın olacaktır. .[7]

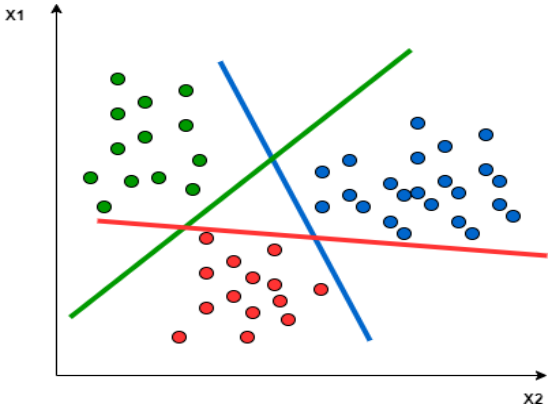
SVM, çok sınıflı sınıflandırmayı aslında desteklemez ; fakat 2 şekilde yeniden yapılandırılabilir.

One vs one style e one vs rest style ;

Biz one vs rest kullandığımız için ona değineceğiz:

Bu yaklaşımda, sınıflandırıcı m adet ikili Destek Vektör Makinesi (SVM) kullanır. Bu m SVMs'lerin her biri, örneğin pozitif sınıf (ilgi alanı olan sınıf) veya negatif sınıf (diğer tüm sınıflar) içinde olup olmadığını tahmin etmek için eğitilir. Tahmin sırasında, m sınıflandırıcıya bir örnek gönderilir ve en yüksek tahmini olasılık değerine sahip sınıf son tahmin olarak seçilir.

Bir sınıf ile diğer tüm sınıflar arasında bir hiperdüzlemin ayrılması gerekir. Bu, noktaların tümünün dikkate alındığı ve iki gruba ayrıldığı anlamına gelir; bir sınıf noktaları için bir grup ve diğer tüm noktalar için bir grup. Örneğin, yeşil çizgi yeşil noktalar ve diğer tüm noktalar arasında mümkün olduğunca fazla ayrımı sağlamaya çalışır.



Şekil 33. Support Vector Machines multiclass example.[8]

Kayıp fonksiyonu olarak hinge loss fonksiyonu kullandık.

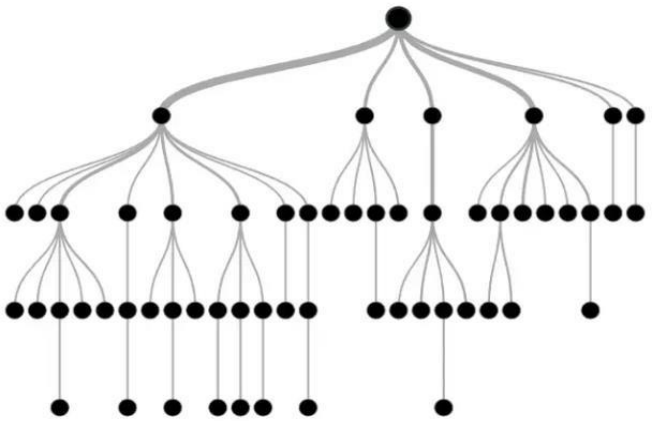
Hinge kayıp fonksiyonu= L(y, f(x)) = max(0, 1 - y * f(x))

y =gerçek etiketimiz(-1,1) aralığında

f(x)=tahmin edilen etiket(-1,1) aralığında

3) DECİSİON TREES(KARAR AĞAÇLARI):

Ağaç tabanlı öğrenme algoritmaları, en çok kullanılan gözetimli öğrenme algorimalarındandır. Genel itibariyle ele alınan bütün problemlerin (sınıflandırma ve regression) çözümüne uyarlanabilirler.Karar agaçları, tesadüfi orman, gradyen güçlendirme (gradient boosting) gibi yöntemler, her türlü veri bilimi problemlerinde yaygın bir şekilde kullanılmaktadırlar.[9]



Şekil 34 . Karar ağacı

Önceden tanımlanmış bir hedef değişkene sahiplerdir. Yapıları itibariyle en tepeden en aşağı inen bir strateji sunmaktadırlar.Bir karar ağacı, çok sayıda kayıt içeren bir veri kümesini, bir dizi karar kuralları uygulayarak daha küçük kümelere bölmek için kullanılan bir yapıdır. Yani basit karar verme adımları uygulanarak, büyük miktarlardaki kayıtları, çok küçük kayıt gruplarına bölerek kullanılan bir yapıdır.

2 tane kayıp fonksiyonu vardır. Gini impurity ve entropy.

Biz default halini scikit librarydeki default halini kullandık o da gini impurity.

Gini impurity:

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

Şekil35. Gini kayıp fonksiyonu formülü

C= sınıf sayısı pi = i inci sınıfın olasılığı

ELM472 Makine Öğrenmesinin Temelleri

4) YAPAY SİNİR AĞI:

Yapay sinir ağları insan beyni örnek alınarak, öğrenme sürecinin matematiksel olarak modellenmesi sonucu ortaya çıkmıştır. Beyindeki biyolojik sinir ağlarının yapısını, öğrenme, hatırlama ve genelleme kabiliyetlerini taklit eder.

Yapay sinir ağlarında öğrenme işlemi örnekler kullanılarak gerçekleştirilir. Öğrenme esnasında giriş çıkış bilgileri verilerek, kurallar koyulur.[10]

Bir yapay sinir hücresi beş bölümden oluşmaktadır;

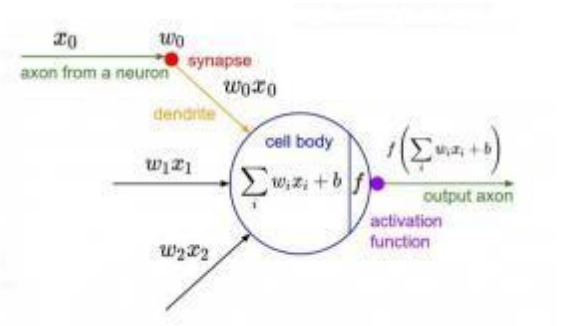
1. Girdiler: Girdiler nöronlara gelen verilerdir. Bu girdilerden gelen veriler biyolojik sinir hücrelerinde olduğu gibi toplanmak üzere nöron çekirdeğine gönderilir.

2. Ağırlıklar: Yapay sinir hücresine gelen bilgiler girdiler üzerinden çekirdeğe ulaşmadan önce geldikleri bağlantıların ağırlığıyla çarpılarak çekirdeğe iletilir. Bu sayede girdilerin üretilecek çıktı üzerindeki etkisi ayarlanabilinmektedir.

3. Toplama Fonksiyonu (Birleştirme Fonksiyonu): Toplama fonksiyonu bir yapay sinir hücresine ağırlıklarla çarpılarak gelen girdileri toplayarak o hücrenin net girdisini hesaplayan bir fonksiyondur

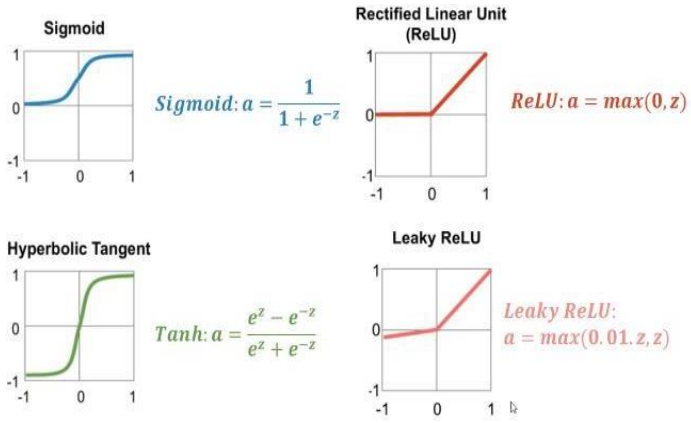
4. Aktivasyon fonksiyonu: Önceki katmandaki tüm girdilerin ağırlıklı toplamını alan ve daha sonra bir çıkış değeri (tipik olarak doğrusal olmayan) üreten ve bir sonraki katmana geçiren bir fonksiyondur. (örneğin, ReLU veya sigmoid).[10]

5. Çıktılar: Aktivasyon fonksiyonundan çıkan değer hücrenin çıktı değeri olmaktadır. Her hücrenin birden fazla girdisi olmasına rağmen bir tek çıktısı olmaktadır. Bu çıktı istenilen sayıda hücreye bağlanabilir.



Şekil 36.Perceptron matematiği şekli

AKTİVASYON FONKSİYONLARI:



Şekil 37. Kullandığımız bazı aktivasyon fonksiyonları formülleri

```
model=Sequential()
model.add(Dense(100,input_shape=(30,)))
model.add(Activation('relu'))
model.add(Dropout(0.5))

model.add(Dense(200))#200 nöron
model.add(Activation('relu'))
model.add(Dropout(0.5))

model.add(Dense(100))#100 nöron
model.add(Activation('relu'))
model.add(Dropout(0.5))


model.add(Dense(num_labels))#5 çıkış
model.add(Activation('softmax'))# soft
```

```
# Model tanım
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 100)               3100

 activation (Activation)     (None, 100)               0

 dropout (Dropout)           (None, 100)               0

 dense_1 (Dense)             (None, 200)               20200

 activation_1 (Activation)   (None, 200)               0

 dropout_1 (Dropout)         (None, 200)               0

 dense_2 (Dense)             (None, 100)               20100

 activation_2 (Activation)   (None, 100)               0

 dropout_2 (Dropout)         (None, 100)               0

 dense_3 (Dense)             (None, 5)                 505

 activation_3 (Activation)   (None, 5)                 0

=================================================================
Total params: 43,905
Trainable params: 43,905
Non-trainable params: 0
_________________________________________________________________
```

Şekil 38.Kullandığımız bir örnek model.

ELM472 Makine Öğrenmesinin Temelleri

Sequential modeli, katmanların düzgün bir sıralı yığınıdır ve modele yeni katmanlar eklemek için .add() kullanılır Dense(100,input_shape=(30,)) ise 30 girişe(feature sayımız) 100 nörona gireceği için 100 çıkış yaratır

Dropout: Özetle hidden ya da input layerdan belli kuralla göre (eşik değeri kullanarak ya da rastgele) belli nodelar kaldırılması tekniğidir. Overfitting engellemek için yapılır.Dropout kullanılarak fully-connected layerlardaki bağlar koparılır. Böylece node'lar birbiri hakkında daha az bilgiye sahip olur ve bunun doğal sonucu olarak node'lar birbirlerinin ağırlık değişimlerinden daha az etkilenirler. Bu nedenle dropout yöntemi ile daha tutarlı (robust) modeller oluşturulabilmektedir. Aynı zamanda her bir katmanda farklı hidden unit kombinasyonları birbiriyle çalıştığı için daha iyi bir öğrenme gerçekleşecektir. Bu anlamda dropout kullanıldığında gizli katmanların random forest gibi ensemble çalıştığı düşünülebilir. Bu durum da model için hem zamandan hemde başarım açısından daha iyi performans sağlayacaktır.

Softmax: Çoğu zaman çoklu Sigmoid olarak da bilinen bu fonksiyon, çok sınıflı hedef değişkeni içeren sınıflandırma problemleri (multi-class classification) için uygun bir aktivasyon fonksiyonudur. Softmax, çıktı olarak her sınıfa ait olasılık sonucu döndürür. Yani Softmax ile bir sinir ağı kuruyorsanız, çıktı katmanınızda hedef değişkeniniz kadar nöron olması gerekli.Bu yüzden son katmanımız 5 nöronlu.

Softmax activation function

$$\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

z bizim sınıf vektörümüz.

Şekil39. Softmax formülü

Confusion Matrix:

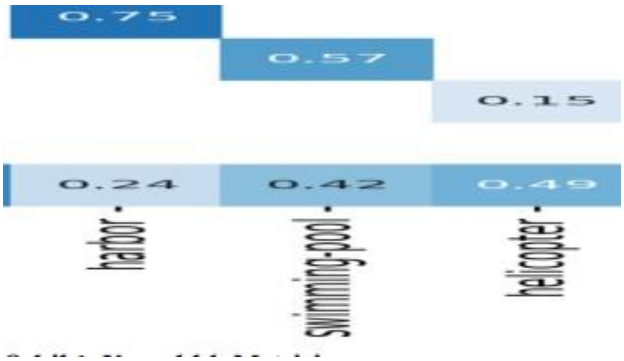
Makine öğrenimi sınıflandırma problemi için bir performans ölçümüdür. Öngörülen ve gerçek değerlerin kombinasyonunu içeren bir tablodur.Karışıklık matrisinde, bilinmesi gereken bazı terimler vardır. Bu terimler doğruluk(accuracy), kesinlik(precision), duyarlılık(recall), gibi çeşitli metrikleri hesaplamak için kullanılabilir. Bu terimler aranan sınıfın "uçak" olduğu durum için şöyle gösterilebilir.

• TP(True Positive – Doğru Pozitif): Uçak olan objeye uçak demek.

• FP (False Positive – Yanlış Pozitif): Uçak olmayan objeye uçak demek.

• TN(True Negative – Doğru Negatif): Uçak olmayan objeye uçak değil demek.

• FN(False Negative – Yanlış Negatif): Uçak olan objeye uçak değil demek.



Şekil 40.Confusion matrix örneği

$$Accuracy = \frac{T_p + T_n}{T_p + T_n + F_p + F_n}$$

$$Precision = \frac{T_p}{T_p + F_p}$$

$$Recall = \frac{T_p}{T_p + T_n}$$

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Şekilde gösterilen grafikte model, harbor (liman) sınıfını 100 örnekten 75'i liman, 24'ünü ise arka plan olarak tahmin etmiştir. Swimming pool (yüzme havuzu) sınıfını 100 örnekten 57'si havuz, 42'si ise arka plan olarak tahmin etmiştir.

Doğruluk(Accuracy): Doğru olarak sınıflandırılan örneklerin yüzdesidir.

Kesinlik(precision): Pozitif olarak tahmin edilen değerlerin kaç tanesinin doğru olduğunu göstermektedir.

Duyarlılık(Recall): Pozitif olarak tahmin edilmesi gereken işlemlerin ne kadarını pozitif olarak tahmin edildiğini göstermektedir.

F1 skoru: Bir testin doğruluğunun bir ölçüsüdür, kesinlik ve duyarlılığın harmonik ortalamasıdır. Makimum 1 ve minimum 0 değerine sahip.

ELM472 Makine Öğrenmesinin Temelleri

Hiperparametreler:

Hiperparametre, bir makine öğrenimi modelinin performansını etkileyen değiştirilebilir değerlerdir.

Öğrenme katsayısı

Nöron sayısı

Epoch sayısı

Batch boyutu

Layer sayısı

Aktivasyon fonksiyonları

Dropout

Üzerinde değişiklik yaptığımız hiperparametreler:

1) Nöron sayısı

2) Aktivasyon fonksiyonu türü

3) Layer sayısı

4)Dropout

Bu hiperparametrede değişiklik yaparak derin öğrenme modelimizi güncelledik ve sınıflandırmayı bu şekilde gerçekleştirdik.

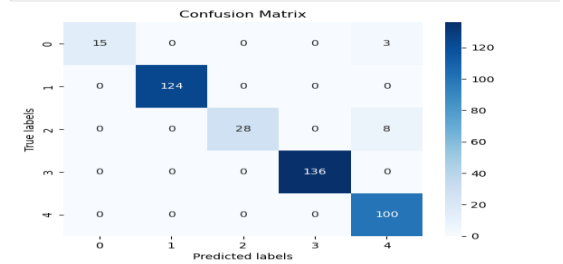
Dengesiz bir veri setiniz varsa ancak veri setinde daha fazla örnek bulunan sınıflara daha fazla katkı atamak istiyorsanız ağırlıklı ortalama(weighted average) tercih edilir.Bunun nedeni, ağırlıklı ortalamada her sınıfın F1 ortalamasına katkısının, sınıfın büyüklüğüne göre ağırlıklandırılmasıdır. [11]

VI)SONUÇLAR VE YORUMLAR

1)K-NEAREST NEİGHBOUR:

	precision	recall	f1-score	support
0	0.79	0.83	0.81	18
1	0.97	0.90	0.93	124
2	0.76	0.78	0.77	36
3	0.91	0.97	0.94	136
4	0.92	0.90	0.91	100
accuracy			0.91	414
macro avg	0.87	0.88	0.87	414
weighted avg	0.91	0.91	0.91	414

Şekil 41.

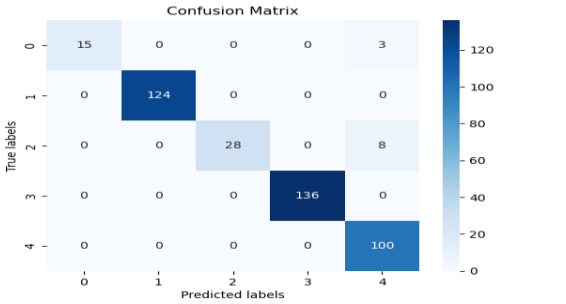


Şekil42.

2) SUPPORT VECTOR MACHİNES:

	precision	recall	f1-score	support
0	1.00	0.83	0.91	18
1	1.00	1.00	1.00	124
2	1.00	0.78	0.88	36
3	1.00	1.00	1.00	136
4	0.90	1.00	0.95	100
accuracy			0.97	414
macro avg	0.98	0.92	0.95	414
weighted avg	0.98	0.97	0.97	414

Şekil43.

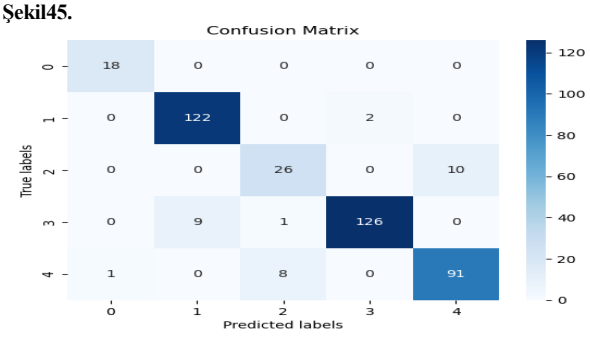


Şekil44.

3) DECİSİON TREES:

	precision	recall	f1-score	support
0	0.95	1.00	0.97	18
1	0.93	0.98	0.96	124
2	0.74	0.72	0.73	36
3	0.98	0.93	0.95	136
4	0.90	0.91	0.91	100
accuracy			0.93	414
macro avg	0.90	0.91	0.90	414
weighted avg	0.93	0.93	0.92	414

Şekil45.



Şekil46.

ELM472 Makine Öğrenmesinin Temelleri

4) MODEL NEURAL 1:

```
In [27]: model=Sequential()# sequential model:
         #input layer
         model.add(Dense(100,input_shape=(30,)))#100 tane nöron ; input shape :30(feature sayımız)
         model.add(Activation('relu'))
         model.add(Dropout(0.5))#ezberlemesini engellemek için unutma katmanı koyuyoruz.

         #first layer
         model.add(Dense(200))#200 nöron
         model.add(Activation('relu'))
         model.add(Dropout(0.5))
         #second layer
         model.add(Dense(100))#100 nöron
         model.add(Activation('relu'))
         model.add(Dropout(0.5))

         #output layer
         model.add(Dense(num_labels))#5 çıkış olmalı 5 sınıf olduğu için
         model.add(Activation('softmax'))# softmax yapıyoruz çünkü sınıfların o olma olasılıksal dağılımını bize veriyor.
```

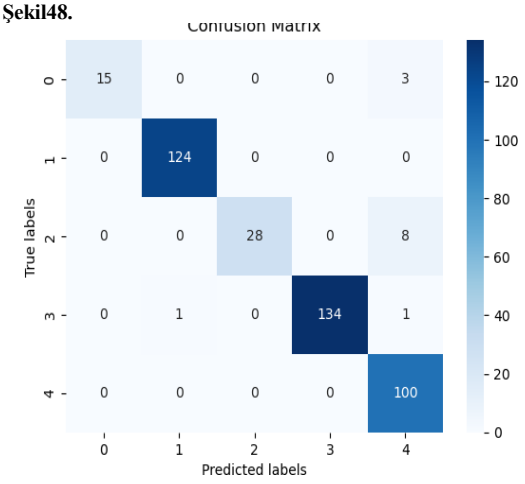
```
In [28]: # Model tanımı
         model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 100)               3100

 activation (Activation)     (None, 100)               0

 dropout (Dropout)           (None, 100)               0

 dense_1 (Dense)             (None, 200)               20200

 activation_1 (Activation)   (None, 200)               0

 dropout_1 (Dropout)         (None, 200)               0

 dense_2 (Dense)             (None, 100)               20100

 activation_2 (Activation)   (None, 100)               0

 dropout_2 (Dropout)         (None, 100)               0

 dense_3 (Dense)             (None, 5)                 505

 activation_3 (Activation)   (None, 5)                 0

=================================================================
Total params: 43,905
Trainable params: 43,905
Non-trainable params: 0
```

Şekil47.

	precision	recall	f1-score	support
0	1.00	0.83	0.91	18
1	0.99	1.00	1.00	124
2	1.00	0.78	0.88	36
3	1.00	0.99	0.99	136
4	0.89	1.00	0.94	100
accuracy			0.97	414
macro avg	0.98	0.92	0.94	414
weighted avg	0.97	0.97	0.97	414

Şekil48.



Şekil49.

MODEL NEURAL 2: relu yerine tanh kullanılmış hali.

```
: model=Sequential()# sequential model:
  #input layer
  model.add(Dense(100,input_shape=(30,)))#100 tane nöron ; input shape :30(feature sayımız)
  model.add(Activation('tanh'))
  model.add(Dropout(0.5))#ezberlemesini engellemek için unutma katmanı koyuyoruz.

  #first layer
  model.add(Dense(200))#200 nöron
  model.add(Activation('tanh'))
  model.add(Dropout(0.5))
  #second layer
  model.add(Dense(100))#100 nöron
  model.add(Activation('tanh'))
  model.add(Dropout(0.5))

  #output layer
  model.add(Dense(num_labels))#5 çıkış olmalı 5 sınıf olduğu için
  model.add(Activation('softmax'))# softmax yapıyoruz çünkü sınıfların o olma olasılıksal dağılımını bize veriyor.
```

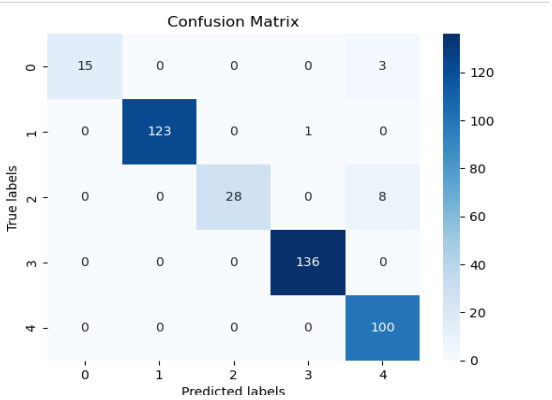
```
: model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_24 (Dense)            (None, 100)               3100

 activation_24 (Activation)  (None, 100)               0

 dropout_18 (Dropout)        (None, 100)               0

 dense_25 (Dense)            (None, 200)               20200

 activation_25 (Activation)  (None, 200)               0

 dropout_19 (Dropout)        (None, 200)               0

 dense_26 (Dense)            (None, 100)               20100

 activation_26 (Activation)  (None, 100)               0

 dropout_20 (Dropout)        (None, 100)               0

 dense_27 (Dense)            (None, 5)                 505

 activation_27 (Activation)  (None, 5)                 0
```

Şekil50.

	precision	recall	f1-score	support
0	1.00	0.83	0.91	18
1	1.00	0.99	1.00	124
2	1.00	0.78	0.88	36
3	0.99	1.00	1.00	136
4	0.90	1.00	0.95	100
accuracy			0.97	414
macro avg	0.98	0.92	0.94	414
weighted avg	0.97	0.97	0.97	414

Şekil51.



Şekil52.

ELM472 Makine Öğrenmesinin Temelleri

MODEL NEURAL 3: 1 layer daha ekledim,dropout ‘u 0.5 ten 0.7 yaptım , nöron sayısını 2 ile çarptım.(output layer hariç)

```
model=Sequential()# sequential model:
#input layer
model.add(Dense(200,input_shape=(30,)))#200 tane nöron ; input shape :30(feature sayımız)
model.add(Activation('tanh'))
model.add(Dropout(0.7))#ezberlemesini engellemek için unutma katmanı koyuyoruz.

#first layer
model.add(Dense(400))#400 nöron
model.add(Activation('tanh'))
model.add(Dropout(0.7))
#second layer
model.add(Dense(200))#200 nöron
model.add(Activation('tanh'))
model.add(Dropout(0.7))
#third layer
model.add(Dense(200))#200 nöron
model.add(Activation('tanh'))
model.add(Dropout(0.7))

#output layer
model.add(Dense(num_labels))#5 çıkış olmalı 5 sınıf olduğu için
model.add(Activation('softmax'))# softmax yapıyoruz çünkü sınıfların o olma olasılıksal dağılımını bize veriyor.
```

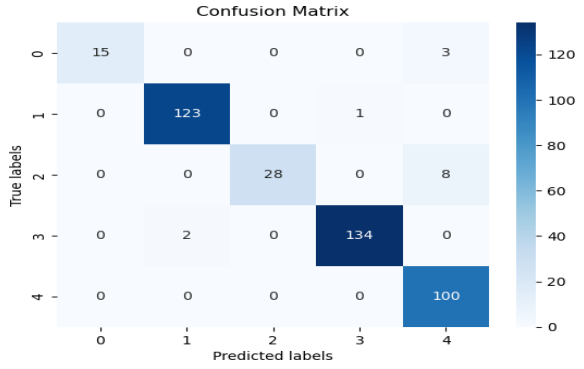
```
model.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_12 (Dense)            (None, 200)               6200

 activation_12 (Activation)  (None, 200)               0

 dropout_9 (Dropout)         (None, 200)               0

 dense_13 (Dense)            (None, 400)               80400

 activation_13 (Activation)  (None, 400)               0

 dropout_10 (Dropout)        (None, 400)               0

 dense_14 (Dense)            (None, 200)               80200

 activation_14 (Activation)  (None, 200)               0

 dropout_11 (Dropout)        (None, 200)               0

 dense_15 (Dense)            (None, 200)               40200

 activation_15 (Activation)  (None, 200)               0

 dropout_12 (Dropout)        (None, 200)               0

 dense_16 (Dense)            (None, 5)                 1005

 activation_16 (Activation)  (None, 5)                 0

=================================================================
Total params: 208,005
Trainable params: 208,005
Non-trainable params: 0
_________________________________________________________________
```

Şekil53.

	precision	recall	f1-score	support
0	1.00	0.83	0.91	18
1	0.98	0.99	0.99	124
2	1.00	0.78	0.88	36
3	0.99	0.99	0.99	136
4	0.90	1.00	0.95	100
accuracy			0.97	414
macro avg	0.98	0.92	0.94	414
weighted avg	0.97	0.97	0.97	414

Şekil54.



Şekil55.

MODEL NEURAL 4: Aktivasyon fonksiyonlarını relu yaptım.

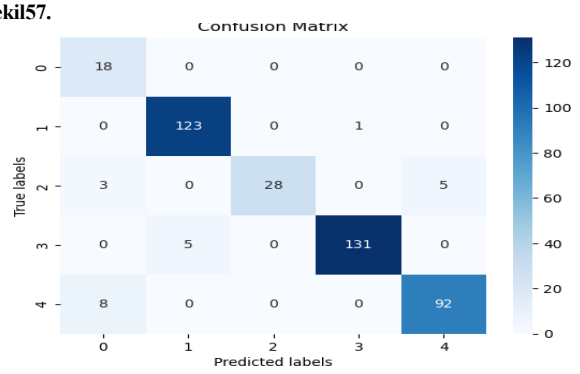
```
#tanh yerine relu kullandım

model=Sequential()# sequential model:
#input layer
model.add(Dense(200,input_shape=(30,)))#200 tane nöron ; input shape :30(feature sayımız)
model.add(Activation('relu'))
model.add(Dropout(0.7))#ezberlemesini engellemek için unutma katmanı koyuyoruz.

#first layer
model.add(Dense(400))#400 nöron
model.add(Activation('relu'))
model.add(Dropout(0.7))
#second layer
model.add(Dense(200))#200 nöron
model.add(Activation('relu'))
model.add(Dropout(0.7))
#third layer
model.add(Dense(200))#200 nöron
model.add(Activation('relu'))
model.add(Dropout(0.7))

#output layer
model.add(Dense(num_labels))#5 çıkış olmalı 5 sınıf olduğu için
model.add(Activation('softmax'))# softmax yapıyoruz çünkü sınıfların o olma olasılıksal dağılımını bize
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_17 (Dense)            (None, 200)               6200

 activation_18 (Activation)  (None, 200)               0

 dropout_14 (Dropout)        (None, 200)               0

 dense_18 (Dense)            (None, 400)               80400

 activation_19 (Activation)  (None, 400)               0

 dropout_15 (Dropout)        (None, 400)               0

 dense_19 (Dense)            (None, 200)               80200

 activation_20 (Activation)  (None, 200)               0

 dropout_16 (Dropout)        (None, 200)               0

 dense_20 (Dense)            (None, 200)               40200

 activation_21 (Activation)  (None, 200)               0

 dropout_17 (Dropout)        (None, 200)               0

 dense_21 (Dense)            (None, 5)                 1005

 activation_22 (Activation)  (None, 5)                 0

=================================================================
Total params: 208,005
Trainable params: 208,005
Non-trainable params: 0
_________________________________________________________________
```

Şekil56.

	precision	recall	f1-score	support
0	0.62	1.00	0.77	18
1	0.96	0.99	0.98	124
2	1.00	0.78	0.88	36
3	0.99	0.96	0.98	136
4	0.95	0.92	0.93	100
accuracy			0.95	414
macro avg	0.90	0.93	0.91	414
weighted avg	0.96	0.95	0.95	414

Şekil57.



Şekil58.

ELM472 Makine Öğrenmesinin Temelleri

MODEL NEURAL 5:ilk layer aktivasyon fonksiyonlarında sigmoid kullandım.İkincide leaky relu kullandım.

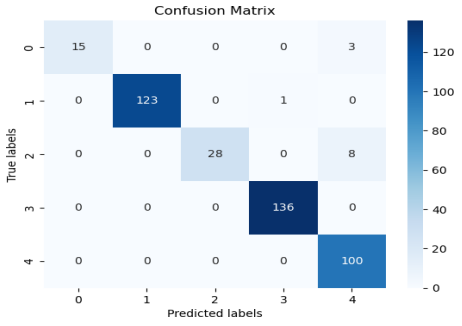
```
model=Sequential()# sequential model:
#input layer
model.add(Dense(200,input_shape=(30,)))#200 tane nöron ; input shape :30(feature sayımız)
model.add(Activation('sigmoid'))
model.add(Dropout(0.7))#ezberlemesini engellemek için unutma katmanı koyuyoruz.

#first layer
model.add(Dense(400))#400 nöron
model.add(tf.keras.layers.LeakyReLU(alpha=0.01))
model.add(Dropout(0.7))
#second layer
model.add(Dense(200))#200 nöron
model.add(Activation('sigmoid'))
model.add(Dropout(0.7))
#third layer
model.add(Dense(200))#200 nöron
model.add(Activation('tanh'))
model.add(Dropout(0.7))

#output layer
model.add(Dense(num_labels))#5 çıkış olmalı 5 sınıf olduğu için
model.add(Activation('softmax'))# softmax yapıyoruz çünkü sınıfların o olma olasılıksal dağılımını
```

```
Model: "sequential_12"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_47 (Dense)            (None, 200)               6200
 activation_43 (Activation)  (None, 200)               0
 dropout_38 (Dropout)        (None, 200)               0
 dense_48 (Dense)            (None, 400)               80400
 leaky_re_lu_5 (LeakyReLU)   (None, 400)               0
 dropout_39 (Dropout)        (None, 400)               0
 dense_49 (Dense)            (None, 200)               80200
 activation_44 (Activation)  (None, 200)               0
 dropout_40 (Dropout)        (None, 200)               0
 dense_50 (Dense)            (None, 200)               40200
 activation_45 (Activation)  (None, 200)               0
 dropout_41 (Dropout)        (None, 200)               0
 dense_51 (Dense)            (None, 5)                 1005
 activation_46 (Activation)  (None, 5)                 0
=================================================================
Total params: 208,005
Trainable params: 208,005
Non-trainable params: 0
```

Şekil59.

	precision	recall	f1-score	support
0	1.00	0.83	0.91	18
1	1.00	0.99	1.00	124
2	1.00	0.78	0.88	36
3	0.99	1.00	1.00	136
4	0.90	1.00	0.95	100
accuracy			0.97	414
macro avg	0.98	0.92	0.94	414
weighted avg	0.97	0.97	0.97	414

Şekil60.



Şekil61.

SONUÇ OLARAK:

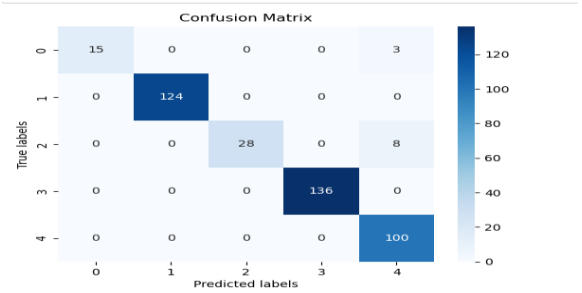
	ACCURACY	WA
KNN	91	91
DT	93	92
SVM	97	97
MODEL 1	97	97
MODEL 2	97	97
MODEL 3	97	97
MODEL 4	95	95
MODEL 5	97	97

```
class_names = ["altı","bebop","savunma","membo","jackal"]
```

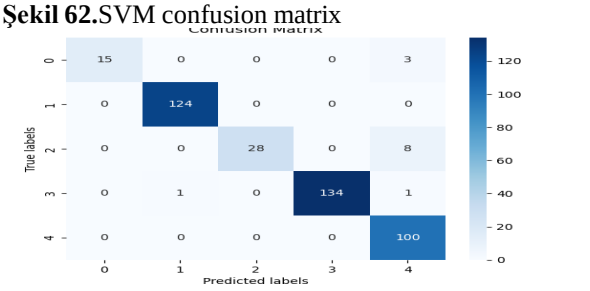
Yorum:

Neural networkler en düşük 95 ile yüksek performans sergiliyorlar ; ama SVM ‘in 97 gibi bir performans sergilemesi benim için şaşırtıcı.

SVM confusion matrix ile Model 1 confusion matrixi karşılaştıralım.



Şekil 62.SVM confusion matrix

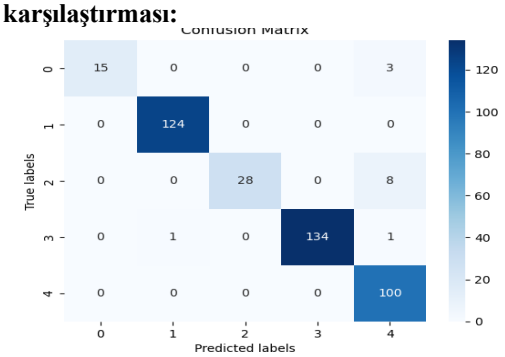


Şekil63. Model 1Neural network confusion matrix .

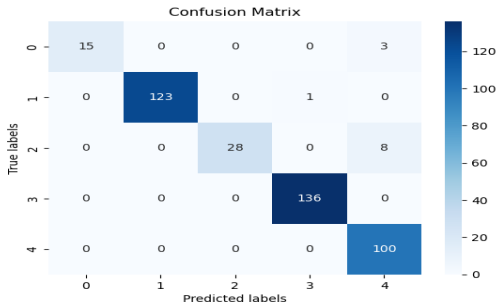
3 numaralı sınıfta görüldüğü üzere SVM aslında daha iyi sonuç vermiş accuracy .2f ile gösterdiğim için 0.97 eşit gibi gözüküyor ;aslında olmadığını neural networkün 3.sınıfta yaptığı 2 hatadan anlayabiliyoruz ; 2 sesten birini 1.sınıf diğerini de 4.sınıf olarak bulmuş fakat aslında öyle değil.

ELM472 Makine Öğrenmesinin Temelleri

Model 1 ile model 2 confusion matrix karşılaştırması:



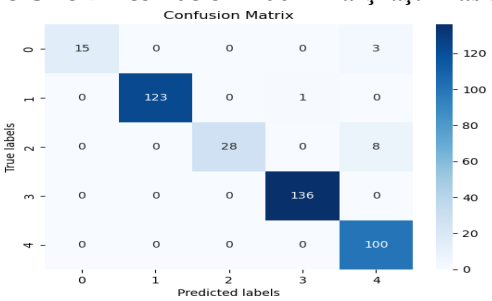
Şekil64. Model 1 confusion matrix



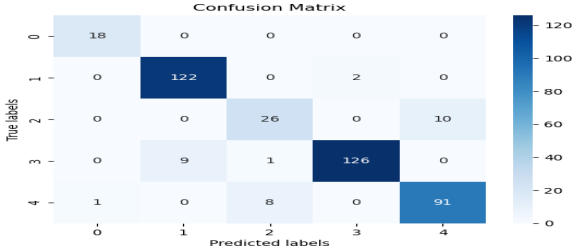
Şekil65. Model 2 confusion matrix

Model 2 birinci sınıfta bir hata yaparken
Model 1 birinci sınıfı hatasız tamamlamış
Model 1 üçncü.sınıfta 2 hata yaparken
Model 2 üçüncü sınıfı hatasız tamamlamış

Model 5 ile DT confusion matrix karşılaştılması:



Şekil66.MODEL 5 Confusion matrix



Şekil67.DT Confusion matrix

DT 0. Sınıfta hepsini biliyor
Model 5 0.sınıfta 5 tane hata yapıyor
DT nin model 5 ten daha düşük accuracy e sahip olmasının sebebi 3. Ve 4. Sınıf ta daha fazla hata yapması.

Sınıflara göre en yüksek başarım gösterenler:

0	MODEL 5
1	MODEL 1,SVM,KNN
2	DT hariç hepsi aynı
3	KNN,SVM,MODEL2,MODEL5
4	MODEL 4 hariç hepsi

Teşekkürler.

Kaynakça:

[1]:https://www.researchgate.net/publication/346243843/figure/fig1/AS:961807523000322@1606324191138/Short-time-Fourier-transform-STFT-overview.png
[2]: https://dergipark.org.tr/tr/download/article-file/202719
[3]: chat gpt
[4]:https://www.youtube.com/watch?v=4_SH2nfbQZ8&t=2216s
[5]: https://stackoverflow.com/questions/16119303/why-dct-transform-is-preferred-over-other-transforms-in-video-image-compression
[6]: https://medium.com/@ekrem.hatipoglu/machine-learning-classification-k-nn-k-en-yak%C4%B1n-kom%C5%9Fu-part-9-6f18cd6185d
[7] : https://medium.com/deep-learning-turkiye/nedir-bu-destek-vekt%C3%B6r-makineleri-makine-%C3%B6%C4%9Frenmesi-serisi-2-94e576e4223e
[8]: https://www.baeldung.com/cs/svm-multiclass-classification
[9]: https://medium.com/@k.ulgen90/makine-%C3%B6%C4%9Frenimi-b%C3%B6l%C3%BCm-5-karar-a%C4%9Fa%C3%A7lar%C4%B1-c90bd7593010
[10] : https://www.veribilimiokulu.com/yapay-sinir-agiartificial-neural-network-nedir/
[11] : https://towardsdatascience.com/micro-macro-weighted-averages-of-f1-score-clearly-explained-b603420b292f#60ad