T.C.

KOCAELİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

LİSANS TEZİ

MAKİNE ÖĞRENMESİ İLE TAŞ-KAĞIT-MAKAS OYUNU GELİŞTİRME

ALPER TALHA KÜÇÜK

MUSTAFA OKSAY

ÖMER HAMSİ

DANIŞMAN

Doç. Dr. PINAR ONAY DURDU

KOCAELİ 2022

T.C.

KOCAELİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

LİSANS TEZİ

MAKİNE ÖĞRENMESİ İLE TAŞ-KAĞIT-MAKAS OYUNU GELİŞTİRME

ALPER TALHA KÜÇÜK

MUSTAFA OKSAY

ÖMER HAMSİ

Doç. Dr. PINAR ONAY DURDU, Danışman, Kocaeli Üniv. …….

Prof. Dr. AHMET SAYAR, Jüri Üyesi, Kocaeli Üniv. …….

Arş. Gör. KÜBRA ERAT, Jüri Üyesi, Kocaeli Üniv. …….

Tezin Savunulduğu Tarih:

ÖNSÖZ VE TEŞEKKÜR

Bu tez çalışması, taş-kağıt-makas oyunun bir arayüz üzerinde görselleştirilip oyunlaştırılması ve taş-kağıt-makas hareketlerinden oluşan bir veri seti üzerinden tahmin modeli eğitilmesi amacıyla gerçekleştirilmiştir.

Tez çalışmamıza yön veren, ihtiyaç halinde yardım ve desteklerini sunan danışmanımız sayın **Doç. Dr. Pınar ONAY DURDU**'ya teşekkürlerimizi sunarız.

Tez çalışmamız boyunca bilgi ve destekleriyle katkıda bulunan sayın **Arş. Gör. Kübra ERAT**'a da ayrıca teşekkür ediyoruz.

Haziran – 2022 Mustafa OKSAY, Alper Talha KÜÇÜK, Ömer HAMSİ

Bu dokümandaki tüm bilgiler, etik ve akademik kurallar çerçevesinde elde edilip sunulmuştur. Ayrıca yine bu kurallar çerçevesinde kendimize ait olmayan ve kendimizin üretmediği ve başka kaynaklardan elde edilen bilgiler ve materyaller (metin, resim, şekil, tablo vb.) gerekli şekilde referans edilmiş ve dokümanda belirtilmiştir.

Öğrenci No: 180202014

Adı Soyadı: Mustafa OKSAY

İmza:

Öğrenci No: 180202034

Adı Soyadı: Alper Talha KÜÇÜK

İmza:

Öğrenci No: 180202111

Adı Soyadı: Ömer HAMSİ

İmza:

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

Simgeler

α	:	Lagrange değişkeni
b	:	Sapma değeri
φ	:	Açı, (°)
σ	:	Sigma
$\phi()$	:	Uzaya eşleyici fonksiyon
$g()$	:	Aktivasyon fonksiyonu
k	:	Parametre
$K()$	:	Çekirdek işlevi
l	:	Destek Vektörlerin Sayısı
x_i	:	Uzaklık
u	:	Nöron girdisi
w	:	Marjin
y	:	$\{-1,1\}$ arası girdi

Kısaltmalar

ANN : Artificial Neural Networks (Yapay Sinir Ağları)

BFT : Best First Decision Tree (En İyi İlk Karar Ağacı)

CNN : Convolutional Neural Networks (Evrişimli Sinir Ağı)

DFS : Depth First Search (Derin Öncelikli Arama)

DVM : Destek Vektörleri ve Marjin

IEEE : The Institute of Electricaland Electronics Engineers (Elektrik ve Elektronik Mühendisleri Enstitüsü)

KNN : K-Nearest Neighbor (En Yakın Komşular Algoritması)

LDA : Linear Discriminant Analysis (Lineer Diskriminant Analizi)

MLP : Multi Layer Perceptron (Çok Katmanlı Algılayıcı)

OVA : One-vs-All (Bire Karşılık Tamamı)

OVO : One-vs-One (Bire Karşılık Bir)

RF : Random Forest (Rassal Orman Algoritması)

RBF : Radial Basis Function (Radyal Temel Fonksiyonu)

SVM : Support Vector Machine (Destek Vektör Makinesi)

ÖZET

Bu çalışmanın amacı bir veri seti üzerindeki görsel verileri, görüntü işleme ve makine öğrenmesi algoritmaları kullanılarak (KNN, SVM vb.) en iyi tahminlemenin yapılmasını sağlamak ve yapılan tahmin sonuçlarını karşılaştırlmalı olarak görselleştirmektir. Bunun yanı sıra bir web arayüzü ile taş-kağıt-makas oyununu simüle etmektir.

Filtreleme operasyonu görüntü işlemede en çok gerçekleştirilen operasyonlardan biridir. Bu operasyonlar iki boyutlu görüntü dizimizdeki her piksel ve çevresindeki pikseller için tanımlanırlar. Filtreleme operasyonlarının büyük çoğunluğu bir pikselin ve çevresindeki komşu piksellerin uygun başka bir matrisle çarpılması ile gerçekleştirirler.

Makine öğrenmesi algoritmaları, büyük kümeyi temsil eden eğitim verilerini temel alan parametreleri kullanır. Eğitim verileri dünyayı daha gerçekçi bir şekilde temsil etmek için genişledikçe, algoritma daha doğru sonuçlar hesaplar.

Uygulamanın ilk aşamasında, HTML ve CSS dilleri ile taş-kağıt-makas oyununun arayüzü oluşturulmuştur. Her bir hareket, her turda oyuncunun ve bilgisayarın seçimleri ve turların toplam sonuçları görselleştirilmiştir. JavaScript dili ile taş-kağıt-makas oyununun mantıksal işlevlerini yerine getirebilmesi sağlanmıştır.

Görüntü işleme ile eldeki verileri kategorize ederek filtre işlemleri uygulanır. Uygulanan filtreleme sonrası öznitelikler çıkartılır. Çıkartılan öznitelik değerleri data frame içerisine atılır. Öznitelikler etiketlenir ve feature içerisinde saklanır. Daha sonra %80'i train, %20'si test olarak bölünen veriler random_state parametresi ile karıştırılır. Bu işlemle, dizi rastgele alt veri setlerine ve test verilerine dönüştürülür.

Eğitim ve test verilerini farklı sınıflandırma modellerinde kullanarak doğruluk değerleri elde edildi ve bu değerler gösterildi. Test verilerinin doğruluk değerlerinden yararlanılarak normalize edilmiş ve edilmemiş karmaşıklık matrisleri oluşturuldu.

Oluşturulan model, eldeki veri setinin dışındaki görüntü dosyaları üzerinde de tahminleme yapabilmektedir.

Anahtar kelimeler: Makine Öğrenmesi, Görüntü İşleme, Veri Seti, KNN, SVM, CSS.

ABSTRACT

The purpose of this study is to provide the best estimation of visual data on a data set by using image processing and machine learning algorithms (such as KNN, SVM, etc.) and to visualize the estimation results comparatively. Besides, it is to simulate rock – paper – scissors game with a web interface.

Filtering operation is one of the most performed operations in image processing. These operations are defined for each pixel and surrounding pixels in our two-dimensional image array. The majority of filtering operations are performed by multiplying a pixel and its neighboring pixels by another suitable matrix.

Machine learning algorithms use parameters based on training data representing the large set. As the training data expands to represent the world more realistically, the algorithm calculates more accurate results.

In the first stage of the application, the interface of rock – paper – scissors game was created with HTML and CSS languages. Each move, the player's and computer's choices in each round, and the overall results of the rounds are visualized. With the JavaScript language, it is ensured that the rock – paper – scissors game can fulfill its logical functions.

Filter operations are applied by categorizing the data at hand with image processing. After the applied filtering, the features are extracted. Extracted attribute values are thrown into the data frame. Attributes are tagged and stored in the feature. Then, the data, which is divided into 80% train and 20% test, mixed with the random_state parameter. With this process, the array is randomly transformed into subdatasets and test data.

By using the training and test data in different classification models, accuracy values were obtained and these values were shown. Normalized and unnormalized confusion matrices were created using the accuracy values of the test data.

The created model can also make predictions on image files other than the available data set.

Keywords: KNN, SVM, Machine Learning, Data Set, Image Processing, CSS.

GİRİŞ

Makine öğrenmesi; sağlık hizmetleri, eğitim, ulaşım, eğlence ve çok daha fazlasını içeren tüm alanları dönüştürerek yaşamı ve toplumları etkilemektedir. Nesnelerin interneti ve bulut bilişim gibi teknolojilerin tümünde, nesneleri daha akıllı hale getirmek için makine öğrenmesinin uygulamalarından giderek daha çok yararlanılmaktadır.

Makine öğrenmesinin yararlanıldığı alanlardan birisi de görüntü işlemedir. Görüntü verilerinin özelliklerinden yararlanılarak çeşitli öznitelikler çıkarılır. Bu özniteliklerin yardımıyla veri setindeki görüntü verileri etiketlenirler. Bunlardan yararlanılarak oluşturulacak olan bir denetimli öğrenme modeli, görüntü tahminlemesi için kullanılabilir. Bu çalışma, bu şekildeki bir uygulama için örnek niteliği taşımaktadır.

Bu çalışmada şu araştırma soruları için yanıtlar aranmaktadır:

1-Görüntü verilerinden çıkarılabilecek öznitelikler nelerdir?

2-Görüntü verilerine filtreleme uygulanmalı mıdır? Uygulanmalıysa neden uygulanmalıdır ? Görüntü verilerindeki gürültü nasıl giderilir?

3-Hangi sınıflandırma algoritmalarını seçmeliyiz?

4-Kullanılan algoritmalar ile elde edilen doğruluk değerlerinden çıkarılabilecek sonuçlar nelerdir ?

“Temel Kavramlar” başlıklı bölümde; doğal etkileşim, insan-bilgisayar etkileşimi ve EMG sensörleriyle etkileşim alt başlıkları incelenmiştir.

Bu çalışmanın “Temel Kavramlar” başlıklı bölümünde, çalışmanın ilgili olduğu alanlar ve kavramlar açıklanmış ve bunlarla ilgili genel bilgilere yer verilmiştir. “Deney ve Yöntemler” bölümünde, çalışmada kullanılan yöntemler ile ilgili detaylı teknik bilgiler ve açıklamalar yer almaktadır. “Sonuç” bölümünde, çalışmanın sonucunda elde edilmiş olan değerlere yer verilmiş olup, deneyin sonucu ile ilgili çıkarımlar bulunmaktadır.

TEMEL KAVRAMLAR

1. GÖRÜNTÜ İŞLEME

Dijital görüntü işleme en genel anlamıyla, bir bilgisayarın dijital görüntüleri bir algoritma aracılığıyla işlemek için kullanılmasıdır. Girişin bir görüntü olduğu ve çıkışın görüntü veya o görüntü ile ilişkili özellikler olabileceği bir sinyal işleme türüdür. Dijital sinyal işlemenin bir alt alanı olarak dijital görüntü işlemenin analog görüntü işlemeye göre birçok avantajı bulunur. Dijital görüntü işleme, analog görüntü işlemeye kıyasla girdi verilerine çok daha geniş bir algoritma yelpazesinin uygulanmasına izin verir ve işleme sırasında görüntünün bozulması veya gürültü oluşumu gibi sorunları önleyebilir. Görüntüler (en az) iki boyutta tanımlandıklarından, dijital görüntü işleme çok boyutlu sistemler şeklinde modellenebilir. Dijital görüntü işleme tekniği kullanırken her tür verinin geçmesi gereken üç aşama ön işleme, geliştirme ve görüntüleme, bilgi çıkarmadır. Dijital görüntü işlemenin üretimi ve gelişimi temel olarak üç faktörden etkilenir: Birincisi, bilgisayarların gelişimi; ikincisi, matematiğin gelişimi; üçüncüsü çevre, tarım askeri, sanayi, ve tıp gibi bilimlerde çeşitli uygulamalara olan talebin artışıdır. [1]

Görüntü işleme temel olarak aşağıdaki üç adımı içerir:

- Görüntü elde etme araçları aracılığıyla görüntünün içe aktarılması,

- Görüntüyü analiz etme ve manipüle etme,

- Görüntü analizine bağlı olarak sonucun değişeceği bir görüntü veya rapor çıktısı.

2. MAKİNE ÖĞRENMESİ

Makine öğrenmesi, insanların öğrenme şeklini taklit etmek için veri ve algoritmaların kullanımına odaklanan ve doğruluğunu kademeli olarak artıran bir bilgisayar bilimi ve yapay zeka dalıdır. Makine öğrenmesi algoritmaları, yeni çıktı değerlerini tahmin etmek için geçmiş verileri girdi olarak kullanır. Temel olarak makine öğrenmesi, verilerden faydalı tahminler yapmak için 'model' adı verilen bir yazılım parçasını eğitme sürecidir. Bir makine öğrenmesi modeli, bir makine öğrenmesi sisteminin tahmin yapmak için kullandığı veri öğeleri arasındaki matematiksel ilişkiyi temsil eder.

Makine öğrenmesi sistemleri tahmin yapmayı öğrenme şekillerine bağlı olarak üç kategoriye ayrılırlar:

- Denetimli Öğrenme

- Denetimsiz Öğrenme

- Pekiştirmeli Öğrenme

a. Denetimli Öğrenme

Denetimli öğrenme modelleri, doğru cevapları olan çok sayıda veri gördükten ve ardından verideki doğru cevapları üreten öğeler arasındaki bağlantıyı keşfettikten sonra tahminlerde bulunabilir. Bu, tıpkı bir öğrencinin hem soruları hem de cevapları içeren eski sınavları inceleyerek yeni materyal öğrenmesi gibidir. Öğrenci eski sınavlar üzerine yeterince eğitim aldığında, yeni bir sınava girmeye hazır hale gelir. Öğrenmenin 'denetimli' olmasının sebebi, bir insanın makine öğrenmesi sistem verilerini doğru olduğu bilinen sonuçlar ile vermesidir [2].

Denetimli öğrenme için en yaygın kullanım durumlarından ikisi regresyon ve sınıflandırmadır.

b. Denetimsiz Öğrenme

Denetimsiz öğrenme modelleri, herhangi bir doğru cevap içermeyen verileri alarak tahminlerde bulunur. Denetimsiz bir öğrenme modelinin amacı, veriler arasında anlamlı örüntüler belirlemektir. Başka bir deyişle, modelin her bir veri parçasını nasıl kategorize edeceğine dair hiçbir ipucu yoktur. Bunun yerine, modelin kendi kurallarını çıkarması gerekir.

Yaygın olarak kullanılan denetimsiz öğrenme modellerinden biri kümeleme adı verilen bir tekniği kullanır. Model, doğal gruplamaları sınırlayan veri noktalarını bulur. Kümeleme, sınıflandırmadan farklıdır çünkü kategoriler kullanıcı tarafından tanımlanmamıştır.

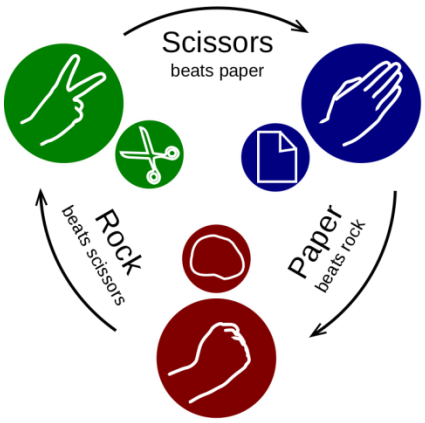
c. Pekiştirmeli Öğrenme

Pekiştirmeli öğrenme modelleri, bir ortamda gerçekleştirilen eylemlere dayalı olarak ödüller veya cezalar alarak tahminlerde bulunur. Pekiştirmeli öğrenme sistemi, en fazla ödülü almak için en iyi stratejiyi tanımlayan bir politika oluşturur.

Pekiştirmeli öğrenme, robotları bir odada dolaştırmak gibi görevleri ve AlphaGo gibi yazılım programlarını eğitmek üzere kullanılır.

3. TAŞ-KAĞIT-MAKAS OYUNU

Taş-kağıt-makas oyunu iki kişiyle oynanan bir oyundur. Oyunda aralarında hiyerarşik bir ilişki bulunan üç adet hareket kullanılır. Oyuncular taş hareketi için yumruk göstermeli, kağıt hareketi için ellerini düz bir biçimde tutarak avuç içini göstermeli, makas hareketi için ise diğer parmaklar kapalı olacak biçimde, işaret ve orta parmaklarını ayrılmış şekilde uzatmalılardır. Her iki oyuncu, yumruk yaptıkları ellerini birbirlerine gösterir biçimde uzatırlar ve seçtikleri hareketi birbirlerine gösterirler. Seçilen hareketler arasındaki ilişkiye göre kazanan belirlenir. Taş hareketi makas hareketine, makas hareketi kağıt hareketine, kağıt hareketi taş hareketine karşı üstündür. Eğer iki oyuncu da aynı hareketi yaparsa oyun tekrarlanır.



Şekil 1. *Taş-kağıt-makas oyunu*

DENEY VE YÖNTEMLER

1. MAKİNE ÖĞRENMESİ

Makine öğrenmesi (ML), bir bilgisayarın doğrudan yönergeler olmadan öğrenmesine yardımcı olmak için matematiksel modelleri kullanma işlemidir. Bu, yapay zekanın (AI) bir alt kümesi olarak kabul edilir. Makine öğrenmesi, verilerdeki kalıpları belirlemek için algoritmaları kullanır. Tahmin yapabilen bir veri modeli oluşturmak için de bu kalıplar kullanılır. Tıpkı insanların daha fazla alıştırma yaptıkça gelişmesi gibi, veri ve deneyim miktarı arttıkça makine öğrenmesinin sonuçları da daha doğru hale gelir [3].

1.1. VERİ TOPLAMA

Makine öğrenmesi algoritmalarının kullanılabilmesi için internet üzerinden edinilen veri seti kullanıldı [4]. Bu veri seti içerisinde farklı kişilerden ve farklı açılardan 840'ar taş, kağıt ve makas görselleri bulunmaktadır. Görsellerin tamamı 300x300 boyutundadırlar ve 24-bit renk değerine sahiptirler. Bu görseller, train, test ve validation başlıkları altında sınıflandırılmıştır.

Şekil 2. *Kağıt veri seti*

Şekil 3. *Taş veri seti*

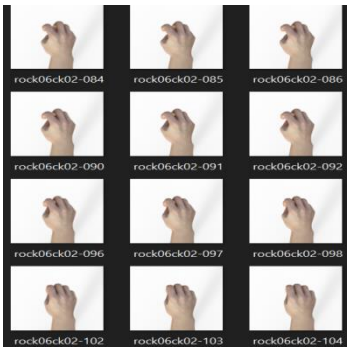
1.2. ÖZNİTELİK ÇIKARIMI (FEATURE EXTRACTION)

Makine öğrenimi, örüntü tanıma ve görüntü işleme alanlarında kullanılan öznitelik çıkarımı (özellik çıkarımı), girdi olarak verilen ölçülmüş verileri kullanarak türetilmiş değerler (öznitelikler) oluşturur. Türetilen değerlerin bilgilendirici ve artıksız olması, öğrenme sürecini kolaylaştırıcı olması ve bazı durumlarda insan uzmanlar tarafından daha iyi anlaşılabilir (yorumlanabilir) olması amaçlanır.

1.3. DENEYSEL ÇALIŞMALAR

A-) Veri Seti

Çalışmada taş-kağıt-makas oyunu için geliştirilmiş geniş bir veri kümesi yer almaktadır. Veri kümesinde, üç hareket için 7 farklı insandan toplamda 2520 adet görüntü elde edilmiştir. Tüm fotoğraflar 300x300 piksel çözünürlüktedirler. Görüntüler elde edilirken zemin, ışık, gölge ve yansıma koşulları sabit tutulmuşlardır.



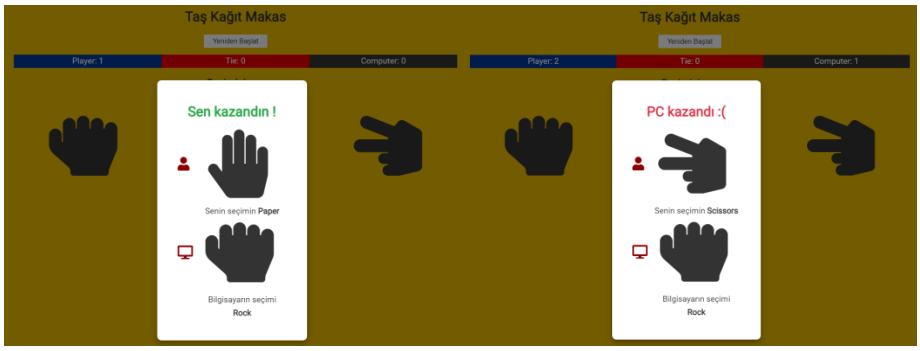
Şekil 4. *Veri seti görüntüleri*

B-) Oyun Yazılımını Geliştirme

Yazılım Javascript programlama dili kullanılarak geliştirilmiştir. Skor tablosu, oyuncu hamleleri, oyun sonuçları gibi bilgiler ve görseller Şekil 6'deki gibi kullanıcı arayüzüne eklenmişlerdir.

Şekil 5.*Kullanıcı arayüzü*

Oyuncu hamlesi ve oyun sonucu ekranda gösterilmiş olup, her biri 10 oyundan oluşan raunt sonuçları ekranın solunda listelenmişlerdir. Raunt içi oyunların skor tablosu ise ekranın üzerinde yer almaktadır.



Şekil 6.*Kazandın ekranı*

Şekil 7.*PC kazandı ekranı*

C-) Sınıflandırma Yöntemleri

Bu çalışmada üç farklı filtre ile öznitelik çıkarımı yapılmış, üç farklı sınıflandırıcının oluşturduğu değerler karşılaştırılmıştır.

C-1) Kullanılan Filtreler

C-1.1) Entropi Kavramı ve Entropi Filtreleme

Entropi, makine öğrenmesinde işlenen bilgilerin rastgeleliğinin veya düzensizliğinin ölçülmesi olarak tanımlanır. Entropi, sistemdeki öngörülemezliği ölçen bir makine öğrenmesi metriğidir. Entropi; öznitelik çıkarımı, karar ağaçları oluşturma ve sınıflandırma modellerinin uyumluluklarının sağlanması gibi işlevleri anlamak için makine öğrenmesinde kullanılan yararlı bir araçtır.

Entropi filtresi ile filtrelenmiş bir görüntüdeki parlak pikseller, daha yüksek entropi değerine sahip komşuluklara karşılık gelmektedirler.

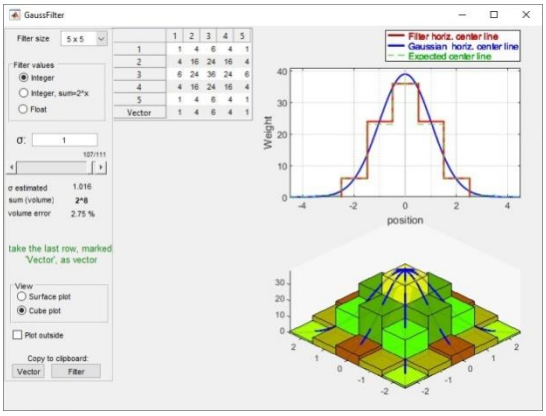
C-1.2) Gauss Filtresi

Gauss filtresi, görüntü üzerindeki gürültüyü kaldırmak ve ayrıntı seviyesini azaltmak için yaygın olarak kullanılan bir filtredir.

$$G(r) = \frac{1}{(2\pi\sigma^2)^{N/2}} e^{-r^2/(2\sigma^2)}$$

Şekil 8.*Gauss filtresinin genel formülü*

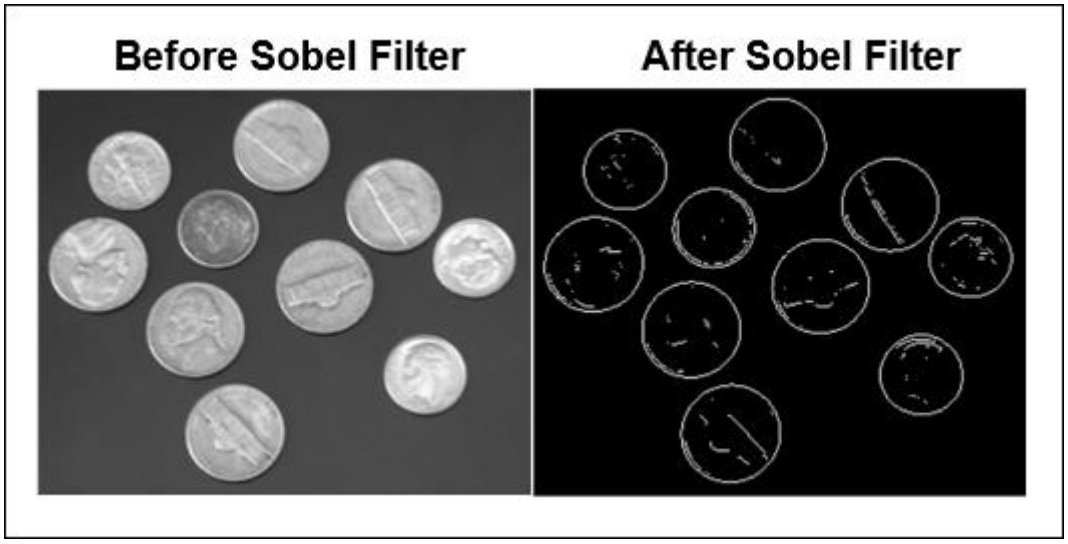
Gauss filtresinde ortalama değere yakın değerlerin grafikteki karşılıkları yüksek iken, ortalama değerden uzaklaşıldıkça karşılık gelen değerler azalırlar. Bu da demektir ki, görüntü işlemede komşuluk mesafesi uzaklaştıkça, komşuluk ilişkiler üzerine yapılan etki azalır. Dolayısıyla Gauss filtresi gerçekleştirilen bir görüntüde ilk olarak komşuluk matrisi oluşturulur. Matris boyutu Gauss fonksiyondaki komşuluk değerine (r) bağlıdır. Matris oluşturulduktan sonra görüntü üzerinde uygulanma işlemi, görüntünün sahip olduğu tüm pikseller üzerinde tek tek işlem yapılarak sağlanmaktadır. Görüntünün (x,y) koordinatlarında alınan piksel değeri, orijinal görüntüdeki komşulara, matristeki katsayılar ile çarpıldıktan sonra yeni değer olarak yazılır. Her piksel, orijinal görüntüdeki komşu piksellerin değerlerini komşuluk yakınlığına bağlı olarak taşımaktadır. Bu durum görüntüdeki ton sertliğinin değişimini azaltmakta ve görüntü üzerindeki gürültünün azalmasını sağlamaktadır.



Şekil 9.*Gauss filtresi uygulanması [5]*

C-1.3) Sobel Filtresi

Sobel filtresi, bir görüntü üzerinde iki boyutlu uzaysal gradyan ölçümü gerçekleştirir ve böylece görüntünün kenarlarına karşılık gelen, en yüksek uzaysal frekans bölgelerini uygular. Gri tonlamalı bir girdi görüntüsünün tüm noktalarında yaklaşık mutlak gradyan değerini bulmak için kullanılır.



Şekil 10.*Sobel filtresi uygulaması[6]*

Sobel filtresi, görüntüdeki kenarların yoğunluk gradyanlarını bulmak için iki boyutta (x ve y boyutlarında) kenar tespiti yapar. Bir görüntüdeki açık ve koyu alanlar arasındaki gradyana uyan bir eğrinin türevini hesaplar ve ardından bir kenar pikselinin konumu olarak yorumlanan türevin tepe noktasını bulur.

1.4. BOYUT İNDİRGEME (DIMENSION REDUCTION)

Boyut indirgeme, öznitelik vektör setindeki değişkenlerin sayılarının azaltılması işlemidir. Boyut indirgeme ile bir verinin yüksek boyutlu bir uzaydan, düşük boyutlu bir uzaya anlamını kaybetmeyecek şekilde dönüştürülmesi amaçlanır. Sınıflandırma problemlerinde, girdilerin sayısını seçmek sınıflandırıcının zaman ve alan

karmaşıklığının hesaplanmasında oldukça büyük bir öneme sahiptir. Bu nedenle daha az boyuta sahip verilerle çalışmak daha basit bir çözüm sağlar.

Boyut indirgeme yöntemleri genel olarak lineer ve lineer olmayan yaklaşımlar şeklinde ikiye ayrılırlar. Boyut indirgeme; veri gürültüsünün (noise) indirgenmesi, verinin görüntülenmesi, küme analizi veya diğer analizleri kolaylaştırmak için bir ara adım olarak kullanılabilir. Sinyal işleme, konuşma tanıma (speech recognition) nöroinformatik, biyoinformatik gibi çok sayıda gözlem ve değişkenle ilgilenen alanlarda kullanımı yaygındır [7].

1.5. SINIFLANDIRMA

Sınıflandırma aşamasında öncelikle öznitelik matrisi eşit sayıda veri içeren eğitim seti(%80) ve test seti(%20) olarak rastgele ikiye bölünmüştür. Sınıflandırılacak modeli eğitmek için eğitim seti ve eğitim modelini test etmek için test seti kullanılmıştır. Eğitim doğrulukları hesaplanırken çapraz doğrulama yöntemi kullanılmıştır ve çapraz doğrulama değeri (cv) için 10 değeri seçilmiştir.

Sınıflandırma yapılırken Destek Vektör Makinesi (SVM), k-En Yakın Komşular (KNN), Karar Ağacı Yapısı ve Lojistik Regresyon yöntemleri kullanılmıştır. SVM, birbirinden ayrılan iki sınıfın verilerinin sınırlarının en uygun şekilde belirlenmesi esasına dayanmaktadır. KNN'de bir test verisine en yakın k verisine bakılır ve hangi sınıfa ait verinin yoğunlaştığı test verisi o sınıfa atanır. Karar ağacı öğrenmesi, öğrenilen fonksiyonun bir karar ağacı tarafından temsil edildiği, kesikli değerli hedef fonksiyonlarını yaklaştırmak için kullanılan bir yöntemdir. Lojistik regresyon, bir sonucu belirleyen bir veya daha fazla bağımsız değişken bulunan bir veri kümesini analiz etmek için kullanılan istatistiksel bir yöntemdir. Sonuç, ikili bir değişkenle ölçülür (yalnızca iki olası sonuç vardır).

1.5.1. k-En Yakın Komşular (KNN) Algoritması:

K-NN *(K-Nearest Neighbor)* algoritması en basit ve en çok kullanılan sınıflandırma algoritmalarından biridir. K-NN parametrik olmayan, tembel bir öğrenme algoritmasıdır. Tembel algoritmalar eğitim aşamasına sahip değillerdir. Eğitim verilerini öğrenemezler, bunun yerine eğitim veri kümesini ‘ezberlerler’. Bir tahmin yapmak istendiğinde, tüm veri setinde en yakın komşuları arar.

Algoritmanın çalışmasında farklı kdeğerleri belirlenir. *k* değerinin anlamı, bakılacak olan eleman sayısıdır. Bir değer geldiğinde en yakın *k* kadar eleman alınarak gelen değer arasındaki uzaklık hesaplanır. Genel olarak, uzaklık hesaplama işleminde Öklid fonksiyonu kullanılır.

Uzaklık hesaplandıktan sonra sıralanır ve gelen değer uygun olan sınıfa atanır.

```
ks = [3,5,7,10]

for k in ks:
    knn = KNeighborsClassifier(n_neighbors=k, metric='euclidean')
    knn.fit(X_train, y_train)
    scores = cross_val_score(
    knn, X_train, y_train, cv=10, scoring='accuracy')
```

Şekil 11. *KNN uygulanması*

Verilen *k* parametresinin ve $D(x, x_i)$ uzaklıklarının sınıflandırma performansı üzerindeki etkisi incelenecektir. Dört uzaklık metriği için 3,5,7 ve 10 artan *k* sayıları test edilmiş ve sonuçlar gösterilmiştir. *k* değerinin azaltılması tüm uzaklık ölçütlerinin doğruluğunu iyileştirmektedir. Optimal komşu sayısı k=3 olarak en iyi uzaklık ölçüsü olarak seçilmiştir.

```
KNN K=3 ile Train Başarısı 91.58730158730158
KNN K=3 ile Test Başarısı 86.34920634920637
KNN K=5 ile Train Başarısı 88.94179894179895
KNN K=5 ile Test Başarısı 85.55555555555556
KNN K=7 ile Train Başarısı 87.98941798941797
KNN K=7 ile Test Başarısı 83.33333333333333
KNN K=10 ile Train Başarısı 86.29629629629629
KNN K=10 ile Test Başarısı 81.42857142857143
```

Tablo 1. *KNN test sonuçları*

1.5.2 Support Vector Machine (SVM) Algoritması:

Popüler öğrenme algoritmalarından biri, temel olarak verileri optimal bir hiper düzlem bularak ayıran SVM'dir. Optimal hiperdüzlem, iki sınıftaki birbirine en yakın noktalar olan destek vektörleri kullanılarak oluşturulur. Hiperdüzlem ve destek vektörleri arasındaki boşluğa marjin (w) denir ve SVM, daha iyi ayırma için maksimum marjini bulmayı amaçlar. SVM'nin optimizasyon problemi şu şekilde tanımlanır:

$$\min_{w} \frac{1}{2}\|w\|^2 \qquad (1.10)$$

Buna bağlı olarak,

$$y_i(w^T\phi(x_i) + b) - 1 \geq 0\,, \forall_i \quad (1.11)$$

$y\epsilon\{-1,1\}$ ve$x$ girdidir, b sapmadır ve $\phi()$ x'i daha yüksek boyutlu bir uzaya eşler. Optimizasyon tamamlandığında, optimal marjin (w) aşağıdakileri karşılar:

$$w = \sum_{i=1}^{l} y_i \alpha_i \, \phi(x_i) \quad (1.12)$$

$l$destek vektörlerinin sayısıdır ve α ise Lagrange değişkenidir. Şimdi, bir sorgu örneği olan x_q geldiğinde, SVM çıktıya şu şekilde karar verir:

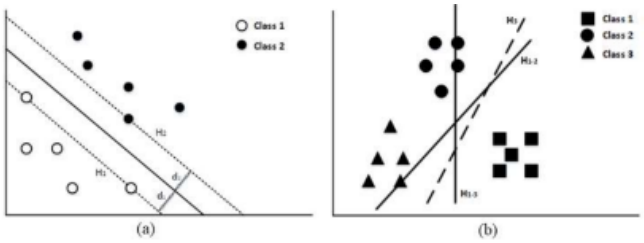
$$f(x_q) = sign\left(\sum_{i=1}^{l} y_i \alpha_i \, K(x_q, x_i) + b\right) \quad (1.13)$$

$K()$, girdi verilerini daha yüksek boyutlu uzaya dönüştürerek SVM'e lineer olmayan özellik sağlayan çekirdek işlevidir. $K()$ aşağıdaki gibi seçildiğinde, doğrusal SVM olarak adlandırılır:

$$K(x_i, x_j) = x_i \cdot x_j \quad (1.14)$$

Ek olarak, bir Radyal Temel Fonksiyonu (RBF) da kullanılabilir.

$$K(x_i, x_j) = \exp(-\frac{|x_i - x_j|^2}{2\sigma^2}) \quad (1.15)$$



Şekil 12.*(a) Destek vektörleri ve margin (b) OVO ve OVA yöntemleri[8]*

Deneyler esnasında, “linear, poly, rbf, sigmoid” şekilleriyle for döngüsüne sokularak train ve test verileri için ayrı olarak doğruluk hesaplaması yapılmıştır.

```
kernels = ['linear', 'poly', 'rbf', 'sigmoid']

for kernel in kernels:
    clf = SVC(kernel=kernel)
    scores = cross_val_score(
        clf, X_train, y_train, cv=10, scoring='accuracy')
```

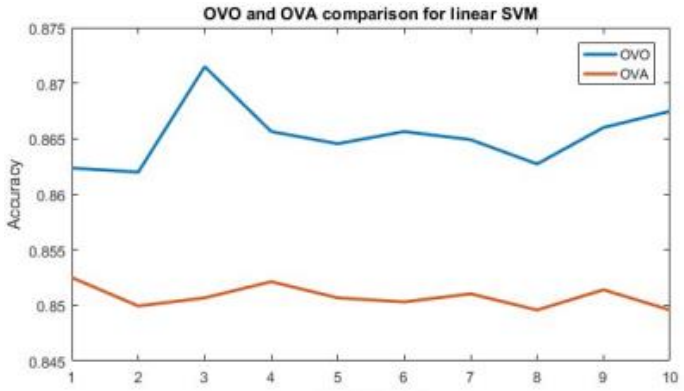
Şekil 13. *SVM uygulanması*

Görülebileceği gibi, SVM Kernel’da poly ile işlenen veri seti için daha iyi bir tanıma oranı sağlanmıştır.

```
SVM Kernel=linear ile train Başarısı 52.85714285714286
SVM Kernel=linear ile test Başarısı 48.57142857142856
SVM Kernel=poly ile train Başarısı 60.63492063492063
SVM Kernel=poly ile test Başarısı 57.93650793650793
SVM Kernel=rbf ile train Başarısı 60.317460317460316
SVM Kernel=rbf ile test Başarısı 54.44444444444445
SVM Kernel=sigmoid ile train Başarısı 28.359788359788357
SVM Kernel=sigmoid ile test Başarısı 26.34920634920635
```

Tablo 2. *SVM test sonuçları*

Belirtildiği gibi, SVM ikili sınıflandırıcıdır, ancak bazı durumlarda veri seti çok sınıflı olabilir. Bu nedenle, ikili sınıflandırıcılardan çok sınıflı çıktılar elde etmek için OVO (bire karşılık bir, İng. One-vs-One) ve OVA (bire karşılık tamamı – İng. One-vs-All) yöntemleri kullanılabilir. OVO, eğitim setindeki her sınıf çifti için ayrı bir sınıflandırıcı eğitir. Öte yandan, OVA, çiftleri seçilen sınıf ve geri kalanı olarak dikkate alarak her sınıf için bir sınıflandırıcı eğitir. Veri seti için OVO ve OVA yöntemlerinin performanslarını karşılaştıran bir grafik aşağıda mevcuttur. OVO yöntemini kullanan lineer SVM'in, OVA karşılığına kıyasla daha yüksek bir doğrulukla sonuçlandığı gözlemlenebilmektedir [şekil 12].



Şekil 14.*OVO-OVA karşılaştırılması[9]*

1.5.3 Karar Ağacı Yapısı:

Karar ağacı, verilerin gruplandırılmasını sağlayan karar verici bir algoritmadır. Bu algoritma ile her bir veri kümesinin sonucunu veri ayrımının bittiği düğümü kontrol ederek tespit etmek mümkündür.

En İyi İlk Karar Ağacı (İng. Best First Decision Tree – BFT) yapısı böl ve yönet mantığına dayanır. Öncelikle bir nitelik kök olarak alınır ve bu nitelik üzerinden entropi kriterine göre gruplamalar yapılır. Daha sonra kök düğümünden itibaren her grup genişletilerek eğitim verileri alt gruplara bölünür. Sadece karşılayan verileri kullanacak şekilde seçilen her grup için bu işlem tekrarlanır. Her işlemde, genişlemeye en uygun alt grup seçilir. Bu süreç belirli bir genişleme katsayısına göre yapılır. Seçilen maksimum genişleme katsayısı 3 olarak alınmıştır. Ağaç derinliği 3 dallanmadan sonra kontrolü gerçekleştirilmemiştir.

```
clf = DecisionTreeClassifier( criterion='entropy',random_state=0,max_depth=3)
clf.fit(X_train, y_train)


scores_train = cross_val_score(
    clf, X_train, y_train, cv=10, scoring='accuracy')

scores_test = cross_val_score(
    clf, X_test, y_test, cv=10, scoring='accuracy')
```

Şekil 15.*Karar ağacı uygulanması*

Karar ağacı kullanmanın avantajları, karar ağacı sonuçlarının yalnızca koşul durumlarını kullanarak kolayca kaynak koda dönüştürülebilmesi ve tüm algoritmayı anlamanın kolay olmasıdır. Dezavantajı ise karar ağacının yalnızca oluşturduğu eğitim verileriyle uyumlu olmasıdır.

1.5.4 Lojistik Regresyon:

Lojistik regresyonda, bağımlı değişken ikili veya ikili, yani yalnızca 1 (DOĞRU, başarı, hamile vb.) veya 0 (YANLIŞ, hata, gebe olmayan vb.) olarak kodlanmış verileri içeriyor. Lojistik regresyonun amacı, iki yönlü karakteristiği (bağımlı değişken = yanıt veya sonuç değişkeni) ile ilgili bir dizi bağımsız (öngörücü veya açıklayıcı) değişken arasındaki ilişkiyi tanımlamak için en uygun (henüz biyolojik olarak makul) modeli bulmaktır [5].

```
reg = LogisticRegression(
    penalty="l1",
    solver="liblinear",
    tol=1e-6,
    max_iter=int(1e6),
    warm_start=True,
    intercept_scaling=10000.0,
)

coefs_ = []
for c in cs:
    reg.set_params(C=c)
    reg.fit(X, y)
    coefs_.append(reg.coef_.ravel().copy())

coefs_ = np.array(coefs_)
reg.fit(X_train, y_train)

scores_train = cross_val_score(
    reg, X_train, y_train, cv=10, scoring='accuracy')
scores_test = cross_val_score(
    reg, X_test, y_test, cv=10, scoring='accuracy')
```

Şekil 16. *Lojistik Regresyon uygulanması*

SONUÇLAR

Proje içerisinde taş-kağıt-makas hareketlerini içeren veri seti kullanılmıştır. Bu veri setleri kategorilere ayrılmıştır. Bu kategorilerden entropi, gaussian ve sobel filtreleriyle data frameler oluşturulmuştur.

	Entropy	Gaussian	Sobel
0	0.669674	0.689635	0.032019
1	0.679112	0.687314	0.032122
2	0.667427	0.681060	0.032306
3	0.698125	0.672518	0.032777
4	0.675442	0.663628	0.033063
...	...	...	...
2515	1.011404	0.682671	0.029446
2516	1.000446	0.682176	0.029372
2517	0.984360	0.682276	0.029387
2518	0.979407	0.682174	0.029397
2519	1.005071	0.682137	0.029506

Tablo 3. *Filtreleme sonuçları*

KNN içerisinde k başlangıç değerleri için 3,5,7 ve 10 değerleri kullanılmıştır. Veri uzaklığı hesabında Öklid metriği kullanılmıştır. Çapraz doğrulama için 10 değeri üzerinden test ve train için doğruluk hesaplanmıştır.

```
KNN K=3 ile Train Başarısı 91.58730158730158
KNN K=3 ile Test Başarısı 86.34920634920637
KNN K=5 ile Train Başarısı 88.94179894179895
KNN K=5 ile Test Başarısı 85.55555555555556
KNN K=7 ile Train Başarısı 87.98941798941797
KNN K=7 ile Test Başarısı 83.33333333333333
KNN K=10 ile Train Başarısı 86.29629629629629
KNN K=10 ile Test Başarısı 81.42857142857143
```

Tablo 4. *KNN başarımları*

SVM algoritmasında "Linear, Poly, RBF, Sigmoid" parametrelerinin her biri için çapraz doğrulama değeri 10 alınarak train-test için doğruluk değeri hesaplaması yapılmıştır.

```
SVM Kernel=linear ile train Başarısı 52.85714285714286
SVM Kernel=linear ile test Başarısı 48.57142857142856
SVM Kernel=poly ile train Başarısı 60.63492063492063
SVM Kernel=poly ile test Başarısı 57.93650793650793
SVM Kernel=rbf ile train Başarısı 60.317460317460316
SVM Kernel=rbf ile test Başarısı 54.44444444444445
SVM Kernel=sigmoid ile train Başarısı 28.359788359788357
SVM Kernel=sigmoid ile test Başarısı 26.34920634920635
```

Tablo 5. *SVM başarımları*

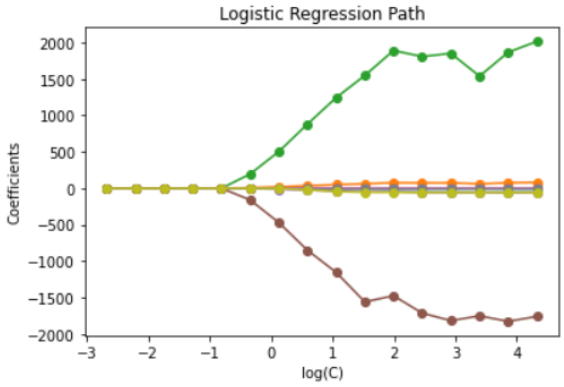
Karar ağacı yapısı için maksimum derinliği 3 olarak belirlenmiş, çapraz doğrulama değeri 10 alınarak train-test için doğruluk değeri hesaplaması yapılmıştır.

```
Decision Train Tree Başarısı = 78.57142857142856
Decision Test Tree Başarısı = 72.85714285714285
KNN Train Başarısı = 91.58730158730158
KNN Test Başarısı = 86.34920634920637
```

Tablo 6. *Karar ağacı başarımları*

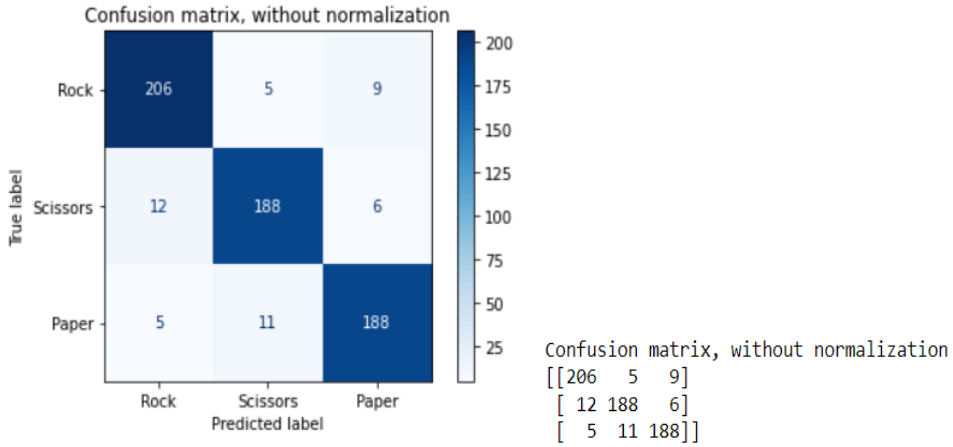
Lojistik regresyon sınıflandırmasında, kurulmuş olan modelin üzerine uygulanan regülarizasyon ile elde edilmiş değerler aşağıdaki gibidir:

```
Logistic Regression = 92.38095238095238
logistic Regression = 92.06349206349206
```



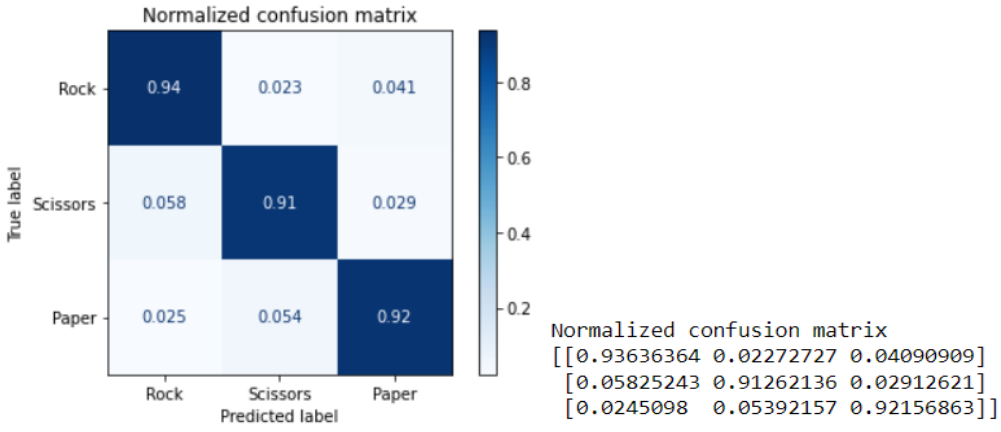
Tablo 7. *Lojistik Regresyon başarımları*

Elde edilmiş karmaşıklık matrisi aşağıdakiler gibi oluşturulmuştur.



Tablo 8. *Normalizasyon uygulanmamış matris*

Karmaşıklık matrisinin normalizasyon işlemi uygulanmış hali aşağıdaki gibi oluşmuştur. Normalizasyon işlemiyle değerler 0-1 aralığına sıkıştırılarak görselleştirilmiştir.



Tablo 9. *Normalizasyon uygulanmış matris*

Model, oluşturulmuş pred klasörü üzerinde filtreleme ve sınıflandırma işlemleri yaparak veri setinin dışında kullanılmış olan bir görsel için de tahmin yapabilmektedir.

Şekil 17. *Veri seti harici görseller*

```
In [12]: clf.predict(df)
Out[12]: array(['paper', 'scissors', 'rock', 'rock'], dtype='<U8')
```

Şekil 18. *Harici verilerin tahminlenmesi*

KAYNAKLAR

[1]- [https://developers.google.com/machine-learning/practica/image-classification]

[2] - [https://developers.google.com/machine-learning/intro-to-ml]

[3] - [https://azure.microsoft.com/tr-tr/overview/what-is-machine-learning-platform]

[4] - [https://www.kaggle.com/datasets/sanikamal/rock-paper-scissors-dataset]

[5] - Peter Seibold (2020). Gaussian Filter, Determination of integer parameters, MATLAB Central File Exchange. Retrieved August 12, 2020

[6] - Sobel Edge Detection Matlab Code Without Using Function, [https://angelabrogan.blogspot.com/2021/07/sobel-edge-detection-matlab-code.html?m=1]

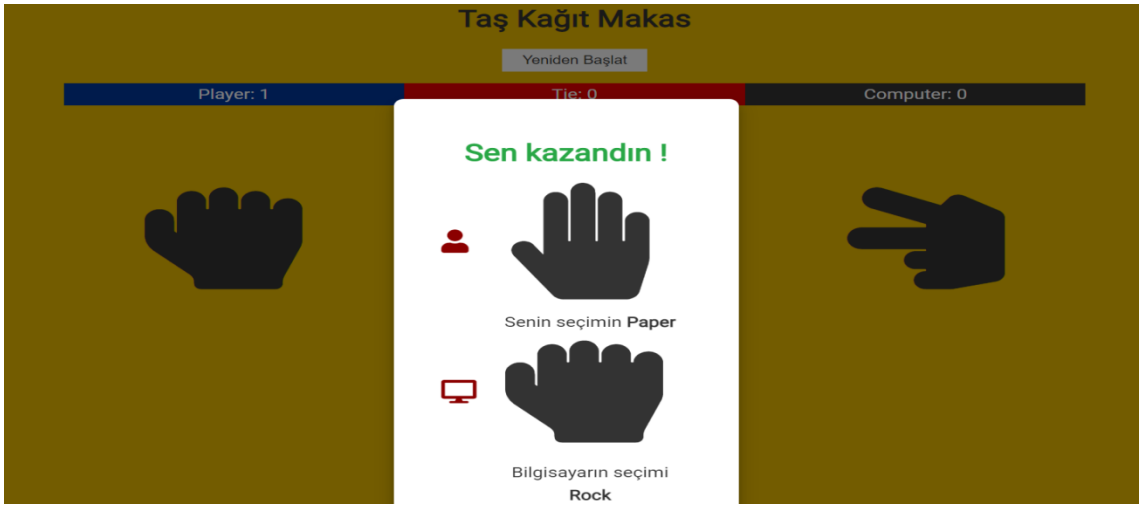
[7]- [https://en.wikipedia.org/wiki/Dimensionality_reduction]

[8] - E. Kaya and T. Kumbasar, "Hand Gesture Recognition Systems with the Wearable Myo Armband," 2018 6th International Conference on Control Engineering & Information Technology (CEIT), 2018, pp. 1-6, doi: 10.1109/CEIT.2018.8751927.

[9] - [https://veribilimcisi.com/2017/07/18/lojistik-regresyon]

EKLER

1. ARAYÜZ



2. HTML KODLARI

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    https://kit.fontawesome.com/3da1a747b2.js
    <link rel="stylesheet" href="CSS/modal.css" />
    <link rel="stylesheet" href="CSS/style.css" />
    <title>Taş-Kağıt-Makas</title>
</head>

<body>

    <div class="container">
        <header class="header">
            <h1>Taş Kağıt Makas</h1>
            <button id="restart" class="restart-btn">Yeniden
Başlat</button>
            <div id="score" class="score">
                <p>Player: 0</p>
                <p>Tie: 0</p>
                <p>Computer: 0</p>
            </div>
        </header>
        <h2>Seçimini yap</h2>
        <div class="choices">
            <i id="rock" class="choice fas fa-hand-rock fa-10x"></i>
            <i id="paper" class="choice fas fa-hand-paper fa-10x"></i>
            <i id="scissors" class="choice fas fa-hand-scissors fa-
10x"></i>
        </div>
    </div>

    <!-- Modal -->
    <div class="modal">
        <div id="result" class="modal-content"></div>
    </div>
    http://JS/main.js


</body>

</html>
```

3. JAVASCRIPT KODLARI

```
const choices = document.querySelectorAll('.choice');
const score = document.getElementById('score');
const result = document.getElementById('result');
const restart = document.getElementById('restart');
const modal = document.querySelector('.modal');
const scoreboard = {
  player: 0,
  tie: 0,
  computer: 0
};

// Play game
function play(e) {
  restart.style.display = 'inline-block';
  const playerChoice = e.target.id;
  const computerChoice = getComputerChoice();
  const winner = getWinner(playerChoice, computerChoice);
  showWinner(winner, playerChoice, computerChoice);
}

// Get computers choice
function getComputerChoice() {
  const rand = Math.random();
  if (rand < 0.34) {
    return 'rock';
  } else if (rand <= 0.67) {
    return 'paper';
  } else {
    return 'scissors';
  }
}
// Get game winner
function getWinner(p, c) {
  if (p === c) {
    return 'draw';
  } else if (p === 'rock') {
    if (c === 'paper') {
      return 'computer';
    } else {
      return 'player';
    }
  } else if (p === 'paper') {
    if (c === 'scissors') {
      return 'computer';
    } else {
      return 'player';
    }
  } else if (p === 'scissors') {
    if (c === 'rock') {
      return 'computer';
```

```
    } else {
      return 'player';
    }
  }
}

function showWinner(winner, playerChoice, computerChoice) {
  if (winner === 'player') {
    // Inc player score
    scoreboard.player++;
    // Show modal result
    result.innerHTML = `
      <h1 class="text-win">Sen kazandın !</h1>
      <i class="fas fa-hand-${playerChoice} fa-10x"></i>
      <p>Senin seçimin <strong>${playerChoice.charAt(0).toUpperCase() +
       playerChoice.slice(1)}</strong></p>
      <i class="fas fa-hand-${computerChoice} fa-10x"></i>
      <p>Bilgisayarın seçimi
<strong>${computerChoice.charAt(0).toUpperCase() +
        computerChoice.slice(1)}</strong></p>
    `;
  } else if (winner === 'computer') {
    // Inc computer score
    scoreboard.computer++;
    // Show modal result
    result.innerHTML = `
      <h1 class="text-lose">PC kazandı :(</h1>
     <i class="fas fa-hand-${playerChoice} fa-10x"></i>
      <p>Senin seçimin <strong>${playerChoice.charAt(0).toUpperCase() +
       playerChoice.slice(1)}</strong></p>
      <i class="fas fa-hand-${computerChoice} fa-10x"></i>
      <p>Bilgisayarın seçimi
<strong>${computerChoice.charAt(0).toUpperCase() +
        computerChoice.slice(1)}</strong></p>
    `;
  } else {
    scoreboard.tie++;
    result.innerHTML = `
      <h1>Berabere</h1>
      <i class="fas fa-hand-${playerChoice} fa-10x"></i>
      <p>Senin seçimin <strong>${playerChoice.charAt(0).toUpperCase() +
       playerChoice.slice(1)}</strong></p>
      <i class="fas fa-hand-${computerChoice} fa-10x"></i>
      <p>Bilgisayarın seçimi
<strong>${computerChoice.charAt(0).toUpperCase() +
        computerChoice.slice(1)}</strong></p>
    `;
  }
  // Show score
  score.innerHTML = `
    <p>Player: ${scoreboard.player}</p>
```

```
    <p>Tie: ${scoreboard.tie}</p>
    <p>Computer: ${scoreboard.computer}</p>
    `;

  modal.style.display = 'block';
}

// Restart game
function restartGame() {
  scoreboard.player = 0;
  scoreboard.tie = 0;
  scoreboard.computer = 0;
  score.innerHTML = `
    <p>Player: 0</p>
    <p>Tie: 0</p>
    <p>Computer: 0</p>
  `;
}

// Clear modal
function clearModal(e) {
  if (e.target == modal) {
    modal.style.display = 'none';
  }
}

// Event listeners
choices.forEach(choice => choice.addEventListener('click', play));
document.addEventListener("keyup", function (event) {
  restart.style.display = 'inline-block';

  let playerKey = "";

  if (event.key === 'a') {
    playerKey = 'rock';
  }
  else if (event.key === 'b') {
    playerKey = 'paper';
  }
  else if (event.key === 'c') {
    playerKey = 'scissors'
  }
  if(['rock','paper','scissors'].includes(playerKey)){
    const computerChoice = getComputerChoice();
    const winner = getWinner(playerKey, computerChoice);
    showWinner(winner, playerKey, computerChoice);
  }
});
window.addEventListener('click', clearModal);
restart.addEventListener('click', restartGame);
```

4. CSS STİL KODLARI

```
@import url('https://fonts.googleapis.com/css?family=Roboto');

:root {
  --primary-color: #003699;
  --red-color: #e60000;
  --dark-color: #333333;
  --light-color: #f4f4f4;
  --lose-color: #dc3545;
  --win-color: #28a745;
  --modal-duration: 1s;
}

* {
  box-sizing: border-box;
  padding: 0;
  margin: 0;
}

body {
  font-family: 'Roboto', sans-serif;
  line-height: 1.6;
  background: #e6b800;
  color: #333;
}

.container {
  max-width: 1100px;
  margin: auto;
  overflow: hidden;
  padding: 0 2rem;
  text-align: center;
}

.restart-btn {
  display: none;
  background: var(--light-color);
  color: #333;
  padding: 0.4rem 1.3rem;
  font-size: 1rem;
  cursor: pointer;
  outline: none;
  border: none;
  margin-bottom: 1rem;
}

.restart-btn:hover {
  background: var(--primary-color);
  color: #fff;
}
```

```css
.header {
  text-align: center;
  margin: 1rem 0;
}

.header h1 {
  margin-bottom: 1rem;
}

.score {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  font-size: 1.2rem;
  color: #fff;
}

.score p:nth-child(1) {
  background: var(--primary-color);
}

.score p:nth-child(2) {
  background: var(--red-color);
}

.score p:nth-child(3) {
  background: var(--dark-color);
}

.choices {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-gap: 2rem;
  margin-top: 3rem;
  text-align: center;
}

.choice {
  cursor: pointer;
}

.choice:hover {
  color: var(--primary-color);
}

.text-win {
  color: var(--win-color);
}

.text-lose {
  color: var(--lose-color);
}
```

```
@media (max-width: 700px) {
  .choice {
    font-size: 110px;
  }
}

@media (max-width: 500px) {
  .choice {
    font-size: 80px;
  }
}
```

5.MODAL CSS KODLARI

```
.modal {
  display: none;
  position: fixed;
  z-index: 1;
  left: 0;
  top: 0;
  height: 100%;
  width: 100%;
  overflow: auto;
  background: rgba(0, 0, 0, 0.5);
}

.modal-content {
  background: #fff;
  text-align: center;
  margin: 10% auto;
  width: 350px;
  border-radius: 10px;
  padding: 3rem;
  box-shadow: 0 5px 8px 0 rgba(0, 0, 0, 0.2), 0 7px 20px 0 rgba(0, 0,
0, 0.17);
  animation-name: modalopen;
  animation-duration: var(--modal-duration);
}

.modal-content h1 {
  margin-bottom: 1rem;
}

.modal-content p {
  font-size: 1.2rem;
  margin-top: 1rem;
}

@keyframes modalopen {
  from {
    opacity: 0;
  }

  to {
    opacity: 1;
  }
}
```