
GAZİ ÜNİVERSİTESİ
Gazi
1926

YAPAY ZEKA TABANLI HAVA HEDEFLERİNİN SINIFLANDIRILMASI

Kadir BAŞKAYA

YÜKSEK LİSANS TEZİ
ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARALIK 2024

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Kadir BAŞKAYA

13/12/2024

YAPAY ZEKA TABANLI HAVA HEDEFLERİNİN SINIFLANDIRILMASI

(Yüksek Lisans Tezi)

Kadir BAŞKAYA

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Aralık 2024

ÖZET

Bu tez, gözetleme radarlarında yapay zekâ tabanlı sınıflandırma yöntemlerinin etkinliğini incelemektedir. Gözetleme radarları, hava trafiğini güvenli bir şekilde yönetmek, hava araçlarının konumlarını belirlemek ve potansiyel tehditleri erken tespit etmek amacıyla kritik bir rol oynar. Çalışmada, Radar Kesit Alanı (RKA), hız ve irtifa gibi uçuş verilerinin analiz edilmesi ve bu verilere dayalı sınıflandırma işlemlerinin gerçekleştirilmesi hedeflenmiştir. Makine öğrenmesi ve derin öğrenme algoritmaları kullanılarak farklı sınıflandırma yöntemlerinin başarı oranları karşılaştırılmıştır. MATLAB ortamında gerçekleştirilen uygulamalarda, Topluluk Torbalama Ağaçları (Ensemble Bagged Trees) ve Ağırlaştırılmış En Yakın Komşular (Weighted KNN) gibi makine öğrenmesi ile CNN ve RNN gibi derin öğrenme algoritmalarına ait performans ölçütleri analiz edilmiştir. Elde edilen bulgular, gözetleme radarlarının hava hedeflerini daha güvenilir ve hızlı bir şekilde sınıflandırmasını sağlamak amacıyla, yapay zekâ temelli modellerin uygulanabilirliğini ortaya koymakta olup hava sahası güvenliği ve etkin hava trafik yönetimi için gelecekteki radar sınıflandırma sistemlerinde yapay zekâ temelli çözümlerinin önemini ve potansiyelini vurgulamaktadır.

Bilim Kodu	:	93438
Anahtar Kelimeler	:	Hava hedeflerinin sınıflandırması, makine öğrenmesi, derin öğrenme, hava gözetleme radarı, hava trafik yönetimi
Sayfa Adedi	:	79
Danışman	:	Prof. Dr. Erol KURT

CLASSIFICATION OF AIR TARGETS BASED ON ARTIFICIAL INTELLIGENCE

(M. Sc. Thesis)

Kadir BAŞKAYA

GAZI UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

December 2024

ABSTRACT

This thesis investigates the effectiveness of AI-based classification methods in surveillance radars. Surveillance radars play a critical role in safely managing air traffic, determining the location of aircraft, and early detection of potential threats. The study aims to analyze flight data such as Radar Cross Section (RCS), speed, and altitude, and to perform classification operations based on these data. The success rates of different classification methods are compared using machine learning and deep learning algorithms. In the applications carried out by MATLAB application, the performance metrics of machine learning algorithms such as Ensemble Bagged Trees and Weighted KNN and deep learning algorithms such as CNN and RNN are analyzed. The findings reveal the applicability of AI-based models to enable surveillance radars to classify air targets more reliably and quickly, and emphasize the importance and potential of AI-based solutions in future radar classification systems for airspace security and effective air traffic management.

Science Code	:	93438
Key Words	:	Aerial target classification, machine learning, deep learning, air surveillance radar, air traffic management
Page Number	:	79
Supervisor	:	Prof. Dr. Erol KURT

TEŞEKKÜR

Yüksek lisans çalışması sürecinde bilgi ve tecrübesiyle bana rehberlik eden değerli hocam Prof. Dr. Erol KURT'a, tez çalışmamın her aşamasında destek olan ve paylaştığım zorluklarda yanımda olan değerli abim İlhan AYDIN'a ve bu yolda beni destekleyen Mustafa ŞAHİN'e teşekkür ederim. Ayrıca tez çalışmamı yaparken yapmış olduğu fedakârlık ve vermiş olduğu emekler ile bana güç veren eşim Merve BAŞKAYA'ya ile adını buraya sığdıramayacağım tüm ailem ve dostlarıma sonsuz minnetlerimi sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
σ	RKA
dBsm	Desibel metre kare
Hz	Hertz
m	Metre
m²	Metre kare
R	Korelasyon katsayısı
s	Saniye

Kısaltmalar	**Açıklamalar**
ANN	Artificial Neural Networks-Yapay Sinir Ağları
ASR	Airport Surveillance Radar-Havaalanı Gözetim Radarı
ATM	Air Traffic Management-Hava Trafiği Yönetimi
AUC	Area Under the Curve-Eğri Altındaki Alan
CNN	Convolutional Neural Networks-Evrişimli Sinir Ağı
IFF	Identification Friend or Foe- Dost Düşman Tanıma
İHA	İnsansız Hava Araçları
KNN	K-Nearest Neighbors-En Yakın Komşular
LSTM	Long Short Term Memory-Uzun Kısa Süreli Bellek
MSE	Mean Squared Error-Ortalama Kare Hatası
PDP	Partial Dependence Plot- Kısmi Bağımlılık Grafiği
RADAR	Radio Detection and Ranging-Radyo Tespiti ve Menzil Belirleme
ReLu	Rectified Linear Units-Düzleştirilmiş Doğrusal Birim
RKA	Radar Cross Section -Radar Kesit Alanı
RNN	Recurrent Neural Networks-Tekrarlayan Sinir Ağı
ROC	Receiver Operating Characteristic-Alıcı İşletim Karakteristiği

Kısaltmalar	Açıklamalar
RPM	Revolutions per Minute-Dakikada Dönüş Sayısı
SAR	Synthetic Aperture Radar-Yapay Açıklıklı Radar
SNR	Signal-to-Noise Ratio Sinyal Gürültü Oranı
SVM	Support Vector Machines- Destek Vektör Makinesi
TCAS	Traffic Collision Avoidance System- Çarpışma Önleme Sistemleri

1. GİRİŞ

Stratejik öneme sahip askeri tesislerin güvenliğini sağlamak ve potansiyel tehdit durumlarında erken uyarı elde etmek amacıyla tehdit kaynaklarının zamanında tespit edilmesi ve sınıflandırılması, savunma sistemleri için büyük önem taşır. Hedeflerin doğru sınıflandırılması, askeri operasyonlarda stratejik karar almayı destekler ve tehditlere karşı etkili bir savunma sağlar. Dost-düşman tespiti (IFF - Identification Friend or Foe) sistemleri ile entegre edilen sınıflandırma süreçleri, potansiyel tehditlere karşı daha hızlı tepkiler verilmesi ile birlikte tehditlerin önceliklendirilmesi açısından önem taşır. Sınıflandırma hataları, yanlış tehdit algısına ve/veya tepki süresinde gecikmelere yol açarak askeri karar alma süreçlerinde güvenlik açıklarına sebep oluşturabilir ve özellikle askeri operasyonlarda ciddi stratejik ve operasyonel sonuçlar doğurabilir [1].

"Radio Detection and Ranging" (Radyo Tespiti ve Menzil Belirleme) ifadesinin kısaltmasından türetilen radar kelimesinin fonksiyonel karşılığı, bir hedefin konumunu, hızını ve diğer özelliklerini temassız tespit etmek için elektromanyetik dalgalar kullanan sistem olarak ifade edilebilir. Elektromanyetik dalga yayan ve sonrasında bu dalgaların hedefe çarpıp geri dönmesiyle elde edilen bilgileri işleyen radar üzerinde uzun yıllardır otomasyon ve veri işleme konularında çalışmalar yapılmaktadır. Günümüzde radar teknolojileri, hava tehditlerinin tespiti ve sınıflandırılması amacıyla kapsamlı bir şekilde kullanılmaktadır. Ancak, geleneksel radar analiz yöntemleri artan veri hacmi ve çeşitliliği ile başa çıkmakta yetersiz kalmaktadır [1]. Radar verilerinin analizinde yapay zekâ tabanlı çözümler, daha hızlı ve yüksek doğrulukla sınıflandırma yapılabilmesine olanak sağlar. Yapay zekâ destekli sınıflandırma sistemlerinin, hedeflerin boyutu, hızı, irtifası, manevrası, uçuş profili gibi çeşitli özellikleri kullanarak daha güvenilir sonuçlar üretmesine bağlı olarak hava hedeflerinin sınıflandırılmasında yapay zekâ tabanlı yaklaşımlar giderek daha fazla benimsenmektedir.

Yapay zekâ teknolojileri, artan bilgi işlem gücü ve yenilikçi algoritmalar sayesinde, savunma sanayiinden sivil havacılığa kadar birçok alanda devrim niteliğinde yenilikler sunmakta olup, hava araçlarını hızlı ve güvenilir bir şekilde analiz ederek hava sahası güvenliğine önemli katkılar sağlamaktadır. Geliştirilen yöntemler, hava araçlarının hız, rota ve operasyonel özelliklerine dayalı olarak radar verilerini analiz eder ve giderek yoğunlaşan uçuş trafiğinin düzenlemesine yardımcı olur. Bu teknolojiler, yalnızca askeri değil, sivil hava

trafiği yönetimi (ATM - Air Traffic Management) süreçlerinde de hava sahasının güvenli ve verimli bir şekilde düzenlenmesini sağlayarak çarpışma risklerini ve diğer operasyonel aksaklıkları azaltır [2].

Havacılık kazaları, genellikle insan hatalarından kaynaklanır. Yapay zekâ tabanlı sınıflandırma sistemleri, olası çarpışma durumlarını tespit etmek ve önceden önlem almak için kullanılır. Uçuş sırasında yapay zekâ, hava trafiği verilerini analiz ederek diğer uçakların ve engellerin konumlarını tespit eder ve riskli durumları önceden fark ederek pilotlara ya da hava trafik kontrolörlerine uyarılar gönderir. Böylece, çarpışma önleme sistemleri (TCAS-Traffic Collision Avoidance System) ve diğer güvenlik sistemleri, yapay zekâ destekli sınıflandırma ile daha etkin hale gelir [3].

Sivil havacılıkta, hava sahasına giren her hava aracının hızlı bir şekilde tanımlanması ve sınıflandırılması gerekmektedir. Özellikle tanımlanamayan hava taşıtları veya araçları, sivil hava sahasında ciddi güvenlik tehditleri oluşturabilir. Yapay zekâ tabanlı sistemler, radar ve diğer sensörlerden gelen sinyalleri analiz ederek, hangi hava taşıtlarının sivil havacılık kurallarına uygun olduğunu ve hangilerinin tehdit oluşturabileceğini hızla belirleyebilir [1]. Yapay zekâ tabanlı sınıflandırma, sivil havacılıkta sadece güvenliği değil, aynı zamanda operasyonel verimliliği de artırır. Uçakların rotalarının optimize edilmesi, yakıt tüketiminin azaltılması ve uçuşların zamanında gerçekleşmesi için hava sahasındaki diğer araçların doğru bir şekilde sınıflandırılması gerekir. Yapay zekâ, uçakların rota planlamasında ve hava trafiği yoğunluğuna göre en uygun uçuş rotalarının belirlenmesinde önemli bir rol oynar [2]. Bu, hem havayolu şirketleri için maliyet tasarrufu sağlar hem de çevresel etkileri azaltır.

Son yıllarda sivil ve askeri havacılıkta İnsansız Hava Araçları (İHA) ve otonom hava taşıtlarının kullanımı hızla artmıştır. Bu araçların güvenli bir şekilde entegre edilmesi için, hava sahasındaki tüm hava araçlarının doğru şekilde sınıflandırılması bir zorunluluktur. Yapay zekâ, İHA'ların hava sahasında etkin bir şekilde yönetilmesi ve diğer hava araçlarıyla olası çarpışma risklerinin önlenmesi için kritik bir teknoloji sağlar [4]. Ayrıca, İHA'lar ile ticari uçuşların aynı hava sahasında uyumlu bir şekilde bulunmasını sağlamak için de sınıflandırma sistemleri gereklidir.

Yapay zekâ, bilgisayar sistemlerinin insana benzer şekilde düşünme ve problem çözme yeteneklerine sahip olmasını sağlayan bir bilim dalıdır [2]. Makine öğrenimi ise yapay

zekanın bir alt dalı olup, sistemlerin verilerden öğrenme ve deneyimlerle performansını artırma yeteneğidir [3]. Toplanan veriler, makine öğrenimi modellerinin eğitimi için uygun hale getirilmelidir. Bu aşamada, veri temizleme, normalizasyon ve özellik çıkarımı gibi işlemler gerçekleştirilir. Derin öğrenme ise, çok katmanlı sinir ağları kullanarak büyük miktarda veriyi analiz edebilir ve bu veriler üzerinden öğrenerek sınıflandırma doğruluğunu artırabilir [4].

Yapay zekâ, makine öğrenimi ve özellikle derin öğrenme algoritmaları ile güçlendirilmiş sınıflandırma sistemleri, hava hedeflerini çeşitli özelliklere göre sınıflandırmada önemli avantajlar sunar. Radar sinyallerinin analiz edilmesi, hedefin boyutu, hareket modeli ve radardan gelen geri dönüş sinyalleri gibi unsurların değerlendirilmesi, yapay zekâ tarafından otomatik olarak yapılabilir hale gelmiştir [3].

Yapay zekâ tabanlı sınıflandırma sistemleri, veri kalitesi ve miktarına bağlı olarak performans gösterebilir. Büyük veri, hava savunma sistemlerinde yapay zekanın etkin bir şekilde kullanılabilmesi için kritik bir bileşendir. Radarlar, elektro-optik sistemler ve uydu görüntüleme sistemleri sürekli olarak büyük miktarda veri üretir. Bu verilerin manuel olarak işlenmesi oldukça zor ve zaman alıcıdır. Yapay zekâ, büyük veri setlerinden anlamlı özellikler çıkararak hava hedeflerinin daha hızlı ve doğru sınıflandırılmasını sağlar [3].

Burada sunulan çalışmamızın amacı, bir gözetleme radarı kullanılarak kapsama alanında olabilecek nesnelerin sınıflandırılmasını sağlayacak algoritmaları araştırmak ve farklı modeller arasında yapılacak uygulama neticesinde bir performans karşılaştırma olanağı sunmaktır. Literatürdeki radar verisi ile sınıflandırma çalışmalarında çeşitli yöntemler bulunmaktadır. Bu çalışma kapsamında, yapay zekâ ile hava hedeflerinin sınıflandırılması konusundaki güncel yaklaşımlar, yöntemler ve uygulamalar ele alınmış olup bununla birlikte toplanan veriler MATLAB programı vasıtasıyla işlenmiş ve tasarlanan modellere ait doğrululuk oranları ile işlevsellikleri tartışılmıştır.

Çalışma altı bölümden meydana gelmektedir.

Çalışmanın birinci bölümünde, çalışmanın tanımı, önemi ve kullanım alanları hakkında genel bilgilendirme ile yapay zekâ tekniklerinin hava hedeflerinin sınıflandırılması üzerindeki etkileri ele alınmıştır.

Çalışmanın ikinci bölümünde, literatürde yer alan sınıflandırma çalışmaları, tercih edilen algoritmaları ve analizleri konu alan farklı çalışmalar incelenmiştir.

Çalışmanın üçüncü bölümünde, hava hedeflerine ait verilerin toplanma süreci ve kullanılacak algoritmalar için verilerin nasıl derlendiğine dair bilgiler anlatılmıştır.

Çalışmanın dördüncü bölümünde yapay zekâ uygulamaları ve yaklaşımlar ayrıntılı olarak gösterilmiştir.

Çalışmanın beşinci bölümünde, tez çalışması kapsamında tasarlanan sınıflandırma uygulamaları ele alınmıştır. MATLAB üzerinde yapılan farklı uygulamalar arasında sonuçların ve yöntemlerin karşılaştırılması yapılmıştır.

Çalışmanın altıncı bölümünde ise, tez çalışması kapsamında elde edilen sonuçlar ve gelecekteki yapılabilecek çalışmalar hakkında bilgi sunulmuştur.

2. LİTERATÜR ARAŞTIRMALARI

Etkili bir hava savunma sisteminin tasarımında veya hava trafiği yönetiminde uçuş güvenliğinin sağlanması hem askeri hem de sivil hava sahası açısından kritik öneme sahiptir. Hava hedeflerinin sınıflandırılması, hava araçlarının türlerine göre analiz edilmesine ve radar kapsama alanının optimize edilmesi ile daha güvenli ve etkin bir hava trafiği sağlanmasına katkıda bulunur. Literatürde, radarlardan elde edilen farklı uçuş verilerini işleyerek hedef sınıflandırması yapmak için çeşitli algoritmalar geliştirildiği görülmüştür. Bu algoritmaların performansı büyük oranda kullanılan veri özelliklerine ve sınıflandırma yöntemine bağlıdır.

Hedef sınıflandırmasına ait literatürde, radar verileri üzerinde analiz yapılırken kullanılan parametreler ve yöntemler açısından geniş bir yelpazede çalışmalar bulunmaktadır. Çalışmalarda, sınıflandırmanın doğruluğunu artırmak için hız, ivme, manevra kabiliyeti gibi kinematik özelliklerin yanı sıra, Sinyal Gürültü Oranı (SNR) gibi RF karakteristik özellikler de dikkate alınmaktadır. Bu özellikler, hedeflerin davranışlarının daha doğru bir şekilde tanımlanmasını ve sınıflandırma algoritmalarının başarısının artırılmasını sağlamaktadır. Genel olarak literatür çalışmalarına bakıldığında, kinematik özellikler ilk tercih edilen özellikler arasında iken doppler izlerinin görüntü işleme teknikleri de yaygın olarak kullanıldığı görülmektedir. Sahadan elde edilen gerçek test verilerinin kullanıldığı çalışmalar, bu tür algoritmaların pratikte uygulanabilir olduğunu ve operasyonel başarıyı artırabileceğini kanıtlamaktadır.

Yapılan bir çalışmada, hız, ivme, dönüş hızı ve doppler gibi kinematik özellikler ile Sinyal Gürültü Oranı (SNR) gibi RF karakteristik özellikler kullanılarak derin öğrenmeye (CNN-Convolutional Neural Networks) dayalı sınıflandırma algoritmalarıyla İHA, İHA olmayan ve mikro İHA taşıtlarına ilişkin sınıflandırmalarının yapılabileceği gösterilmiştir [5]. Sahadan alınan gerçek test verilerinin işlenmesi, yapılan çalışmanın etkinliğini ortaya koymaktadır. Farklı bir çalışmada, kuşların ve insansız hava araçlarının alçak irtifa hava sahasında hareket karakteristikleri, ortalama hızı ve manevra profil faktörleri kullanılarak tespit edilmesinin ardından, makine öğrenme yöntemlerinden biri olan rastgele orman modeli kullanılarak sınıflandırma yapılabileceği açıklanmıştır [6]. Kinematik özelliklerin dışında, en çok kullanılan özelliklerden biri de Radar Kesit Alanıdır (RKA- Radar Cross Section). Havadaki nesneleri yeterli doğrulukla sınıflandırmak için, farklı çalışmalarda veri seti olarak kullanılan görsel kaynakların dışında bir kaynak olarak kabul edilen ve önemli

bir radar imzası olan RKA değerleri kullanılarak makine öğrenme algoritmalarının başarı oranları araştırılmıştır. Yazarlar, farklı insansız hava araçlarının kendi metodolojilerine göre iyi bir doğrulukla sınıflandırılabileceğini bulmuşlardır [7]. Literatürde insansız hava araçları üzerine birçok çalışma bulunmaktadır. Havadaki insansız hava aracından yansıyan yankıları analiz ederek ve RKA gibi özellikleri ayırıcı olarak kullanarak diğer hava nesnelerinden ve kendi aralarından ayırt edilebilir olduğu gösterilmiştir [8,9]. İki savaş uçağının ayırt edilebilirliği, RKA veri tabanı üzerinden Yapay Sinir Ağları (ANN-Artificial Neural Networks) ile hedef sınıflandırması yapılarak gösterilmiştir [10]. Çalışmalardan birinde Kaya ve Kaplan, drone, gemi ve kuş hedeflerinden toplanan radar verileri ışığında sinir ağı algoritmaları kullanarak bir sınıflandırma çalışmasını yüksek başarı oranı ile tamamlamıştır [11]. Başka bir çalışmada, Chen ve meslektaşları farklı makine öğrenme algoritmalarını analiz ederek dağıtım sistemi ekipmanlarında bulunan arızaların kısmi deşarj kusurlarının tespit edilebileceği açıklamıştır [12]. Sentetik Açıklıklı Radar (SAR) tabanlı denizcilik hedef sınıflandırmasında yapay zekanın potansiyelini gösteren önemli bir uygulama olan çalışmada Bentes ile arkadaşları, uydu kaynaklı SAR görüntülerini kullanılarak gemi sınıflandırması için derin öğrenme tabanlı çok çözünürlüklü giriş kanalları kullanan Evrişimli Sinir Ağı (CNN-Convolutional Neural Networks) modeli geliştirmiş ve uydu görüntüleri ile beş sınıfın (yük gemisi, tanker, rüzgar türbini, platform ve liman yapısı) ayrımının yapılabilirliği gösterilmiştir. Ayrıca, geleneksel SVM (Destek Vektör Makinesi-Support Vector Machines) tabanlı sınıflandırıcılarla karşılaştırıldığında, CNN modellerinin özellikle yük gemisi ve tanker gibi benzer sınıflar arasında daha yüksek doğruluk sağladığı belirtilmiştir [13]. Gomez ve arkadaşları, radar tabanlı hedef sınıflandırmasını mikro-doppler özellikleri kullanarak yapmayı hedeflemiştir. Bu çalışmada, dronlar, kuşlar ve insanların radar verileri üzerinden sınıflandırılması üzerine odaklanılmıştır. Yüksek frekanslı ve yüksek çözünürlüklü radar imzaları kullanılarak, hedefin belirli mikro salınım (dönme, titreme gibi) hareketleri analiz edilmiştir. Hedefin mikro-doppler görüntüleri, Kısa Zaman Fourier Dönüşümü (STFT) sonrasında işlenerek sınıflandırma yapılmaktadır. Derin öğrenme tabanlı bir Derin Evrişimli Sinir Ağı (DCNN) modeli vasıtasıyla, insan yürüme ve koşma hareketleri, altı farklı dron modeli ile kuş türleri gibi geniş bir hedef kitlesi üzerinde yüksek doğrulukla sınıflandırma gerçekleştiren ve bu sayede güvenlik tehditleri oluşturan yetkisiz nesneleri belirlenmesinde kullanılabilecek bir yöntem önerilmiştir. Ayrıca, radar sınıflandırma yeteneğinin, çalışma frekansı, bekleme süresi, iletilen güç, darbe süresi vb. gibi radar parametrelerine tabi olduğu belirtilmiştir [14]. Başka bir çalışmada Cai ve arkadaşları, otonom sürüşte milimetre dalga (MMW) radar sensörlerinin sınıflandırma

kabiliyetini geliştirmek için dört farklı radar veri türüne dayalı hedef sınıflandırma modeli geliştirmiştir. Çalışmada, istatistiksel RKA bilgisi, dağıtılmış (zaman alanında) RKA yanıtı, 2B menzil-azimut bilgilerine ait radar görüntüleri ve 3B radar görüntüleri kullanılarak oluşturulan veri setleri ANN ve CNN modelleri ile test edilmiş ve aynı zamanda her iki model arasında bir karşılaştırma yapılmıştır. Otonom araçların çeşitli ortam koşullarında özellikle kamera performansının düştüğü durumlarda güvenliğin artırılmasını amaçlayan çalışmada, düşük görünürlük gibi zorlayıcı hava koşullarında dahi güvenilir hedef algılama ve sınıflandırma sağlanabildiği gösterilmiştir. Bu radar hedef sınıflandırma modelleri, özellikle hareketsiz ve hareketli hedefleri hem kısa hem de uzun menzillerde sınıflandırmada başarılı olmuştur. 3B radar görüntüleri gibi gelişmiş radar sistemleri üzerinde en yüksek doğruluk sağlanırken, klasik radar sistemleri ile de istatistiksel RKA bilgisi ile güvenilir sonuçlar elde edilmiştir [15]. Guy Ardon ve meslektaşları, farklı bir çalışmada, hava hedeflerinin sınıflandırılması için ANN tabanlı bir sınıflandırma algoritması geliştirmiştir. RKA zaman serileri kullanılarak, farklı hava araçlarının kendi aralarında ve diğer nesnelerden ayırt edilebileceği gösterilmiştir. Farklı geometrilere sahip hedeflerin özelliklerine bağlı olarak değişkenlik gösteren RKA değerleri, radar denklemine göre doğrudan SNR değerlerinden elde edilmiştir. Çalışma kapsamında, radar yankıları EMD (Empirical Mode Decomposition) ile ayrıştırılmış ve tam bağlantılı bir ANN modeline girdi olarak sunulmuştur. Algoritma, büyük ve çeşitli bir simüle uçuş yörüngeleri kümesinde test edilmiş ve performansı birkaç farklı yöntemle (K-En Yakın Komşular (K-Nearest Neighbors- KNN gibi)) karşılaştırılmıştır. Test sonucunda, farklı hava araçlarının yüksek doğrulukla sınıflandırıldığı görülmüştür. Çalışma, hedef sınıflandırmasında makine öğrenimi yöntemlerinin potansiyelini ortaya koymaktadır [16]. Başka bir çalışmada, Sajjan ve arkadaşları, radar kinematiği (RKA, azimuth ve yükselti) özelliklerini kullanılarak deniz hedef sınıflandırması için makine öğrenimi yaklaşımını tercih etmişlerdir. Altı farklı gemi modelini sınıflandırarak, gemileri korumak ve nakliye kanalları için kılavuzlar sağlamak amacıyla yapılan çalışmada, ANN modeli geliştirilmiştir, Deniz hedeflerinin güvenilir olarak tanımlanması ve sınıflandırılması için kullanılan S-band frekansına ait RKA veri tabanı CATIA modellerinin farklı açılardan simülasyonu neticesinde üretilmiştir. Böylelikle, denizcilik alanında hedef sınıflandırmasında RKA'nın başarılı bir veri kaynağı olduğu gösterilmiştir [17]. Kim ve arkadaşları, diğer bir çalışmada, yine RKA verilerine dayanarak farklı füze modellerinin sınıflandırılmasını amaçlayan 1B CNN-GRU (Gated Recurrent Unit) tabanlı bir ağ önermektedir. Bu çalışma, çok sınıflı füze hedeflerini sınıflandırmak için 1B CNN'in özellik çıkarma yeteneği ile GRU'nun zaman serisi verilerini

sıralı bir şekilde işleme gücünü birleştirerek hibrit bir model ile yüksek doğruluk oranı elde etmiştir. Çalışma, farklı füze modelleri veya aynı rotaya sahip ama farklı tasarımlardaki füzeleri de doğru şekilde ayırt etmiştir. Gürültüye dayanıklı olması ve zaman serisi verilerdeki sınıflandırma performansının geleneksel yöntemlere kıyasla daha iyi olması bu modelin güçlü yönleri olarak öne çıkmaktadır [18]. Danny Holt ve ekibi, havaalanı gözetim radarları (ASR) ile hava hedeflerinin sınıflandırılmasına odaklandıkları çalışmada, geleneksel radar sinyal işleme yöntemlerinin ötesine LSTM (Long Short-Term Memory Layer) tabanlı bir Tekrarlayan Sinir Ağı (Recurrent Neural Networks-RNN) modeli ile radar verilerinde ardışık ilişkiler oluşturarak sınıflandırma yapılabileceğini göstermiştir. Bu çalışma için Cranfield Havalimanı'nda Saab G1X radar sistemi kullanılarak 22 saatlik veri toplanmış, ADS-B ve OpenSky Network gibi kaynaklardan alınan verilerle eğitilen model, dron, pervaneli uçak ve jet uçakları gibi farklı sınıflar arasında sınıflandırma yaptırılmıştır. İlk LSTM katmanı her bir taramada özellikleri çıkarırken, ikinci katman tam bağlantılı katmanlar aracılığıyla nesneler arasında sınıflandırma yapmaktadır. Özellikle, her bir radar taramasındaki ardışık ilişkileri kullanarak hedefleri ayırt etme kabiliyeti, modelin başarısını artırmıştır. Çalışma, hava güvenliği için radar sinyal işleme ve derin öğrenme entegrasyonunun potansiyelini ortaya koymaktadır. Uygulamada, daha küçük drone sınıfı ve pervaneli uçaklar arasında iyi bir farklılaşma olduğunu göstermiş, ancak, model jet ve helikopter sınıflarını tanımakta kısmen zorlanılmıştır [19]. Kumawat ve arkadaşları radar mikro-doppler imzalarına dayalı küçük hava hedeflerinin otomatik sınıflandırılması amacıyla çalışma yürütmüştür. X-bandında çalışan bir sürekli dalga (CW-Continuous Wave) radar kullanılarak düşük RKA değerlerine sahip beş farklı hedefin (iki kanatlı bir rotor, üç kısa kanatlı bir rotor, üç uzun kanatlı bir rotor, dört kanatlı bir rotor, bir biyonik kuş ile iki kanatlı bir rotor ve biyonik kuş) sınıflandırılması amacıyla DIAT-μSAT veri seti geliştirilmiş ve derin öğrenme tabanlı transfer öğrenme tekniklerinden faydalanarak tasarlanan DCNN modelinin yüksek doğruluk oranına ulaştığı gözlemlenmiştir. Özellikle birden fazla hedefin aynı anda (çoklu) çalıştığı durumlarda da yüksek doğruluk oranları elde edilmiştir [20]. Karlsson ve arkadaşlarının çalışmasında ise CNN tabanlı sınıflandırıcıların dron ve diğer kategorideki nesneleri (kuş gibi) tanımlayabilme yeteneğini incelenmiştir. 77 GHz ile çalışan SAAB SIRS 1600 FMCW radarından elde edilen kinematik ve görüntü veri kümesi işlenerek hedefler üzerinde sınıflandırma yapılmıştır. Tasarlanan sınıflandırma algoritması hem gerçek hem de matematiksel olarak oluşturulan sentetik dron modellerine ait veriler ile eğitilmiştir. Bu çalışmada, genel olarak sınıflandırıcıların veri kümesinde yer almayan

nesnelerle başa çıkma yetenekleri üzerinde odaklanılmıştır. Çalışma ayrıca, sinyal gürültü oranına (SNR) bağlı olarak sınıflandırma başarısının değişimini de ele almaktadır [21].

Yapılan literatür araştırmaları, hava hedeflerinin sınıflandırılması için radar verilerine dayalı farklı yaklaşımları, makine öğrenmesi ve derin öğrenme modellerinin önemi ile potansiyelini ortaya koymaktadır.

3. VERİ TOPLAMA

Hem askeri hem sivil havacılıkta hava trafiğini yönetmek, bununla birlikte askeri operasyonlarda düşman unsurların hareketlerini izlemek için kullanılan gözetleme radarlarından elde edilecek uçuş verilerine dayalı bir çalışma yapılmasına karar verilmiştir. Yapılacak uygulama çerçevesinde, radar kapsama alanında karşılaşılabilecek hedeflere ait sınıflar kuş, helikopter, yolcu uçağı, dron, İHA, savaş uçağı ve rüzgâr türbini olarak belirlenmiştir. Yapay zekâ algoritmaları vasıtasıyla ihtiyaç duyulan sınıf ayrımın yapılması için tercih edilen 3 (üç) adet uçuş verisi RKA, hız ve irtifa değerleridir.

RKA, bir nesnenin radar sinyallerine karşılık olarak ne kadar "göründüğünü" ifade eden bir ölçüdür. Bu değer frekansla değişen ama uzaklıktan bağımsız bir büyüklüktür.

RKA, σ, yüzeye etkiyen elektromanyetik alan şiddeti (*incident electric field*) $\mathbf{E_i}$ ve yüzeyden geri saçılan elektromanyetik alan şiddeti (*scattered electric field*) $\mathbf{E_s}$ olmak üzere şu formülle hesaplanır.

$$\sigma = \lim_{r\to\infty} 4\pi r^2 \frac{|\mathbf{E_s}|^2}{|\mathbf{E_i}|^2} = \lim_{r\to\infty} 4\pi r^2 \frac{|\mathbf{W_s}|}{|\mathbf{W_i}|} \tag{3.1}$$

3.1'de yer alan formüldeki temel varsayım $\mathbf{E}_i$ 'nin düzlemsel bir dalga olduğu ve bu yüzden genliğinin mesafeye bağımlı olmadığıdır. $\mathbf{W}_i$; Hedefe etkiyen güç yoğunluğu, $\mathbf{W}_S$; Hedeften radara geri dönen güç yoğunluğu olarak tanımlanır. Uzak alanda, $\mathbf{E}_S$ değeri $1/r$ ile değiştiğinden, RKA değeri de mesafeden bağımsız olur. Radar Kesit Alanı m^2 cinsinden olmasına rağmen geniş bir aralıkta değerler almasından dolayı genellikle dBsm (desibel metre kare) cinsinden verilir. İki birim arasındaki dönüşüm formülü 3.2'de verilmiştir.

$$\sigma\,[dBsm] = 10\log(\sigma[m^2]) \tag{3.2}$$

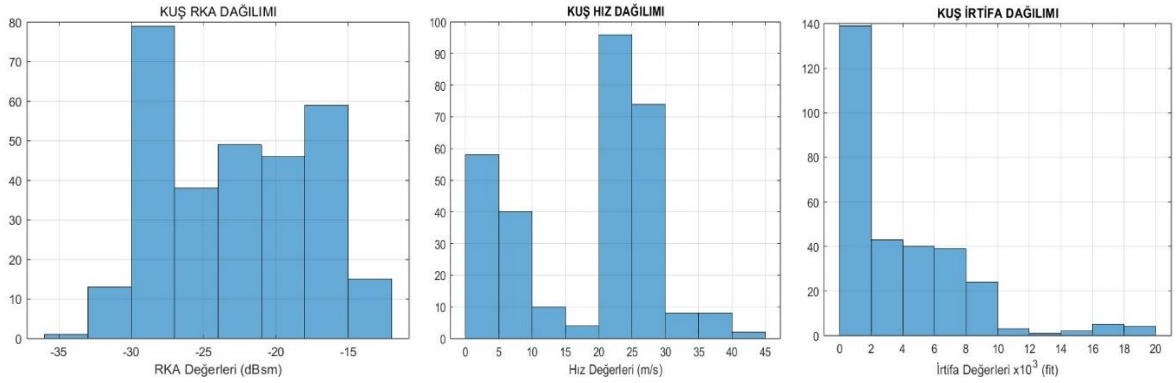
RKA'nın özellikleri, hedefin geometrik şekline, yansıtma katsayısına bağlı olarak malzeme yapısına ve radarın çalışma frekansına göre değişiklik göstermektedir.

Uçuş verilerine ait diğer bir özellik olan hız değerini radar, hedefin konumlarını belirlemesini müteakip zaman içindeki yer değişikliklerini takip ederek hesaplar. Hedefler arasında yer alan rüzgâr türbinine ait hız verisi 3.3'te belirtilen formül ile hesaplanmıştır. RPM pallerin dakikada dönüş sayısını, r ise pal uzunluk değerini karşılamaktadır.

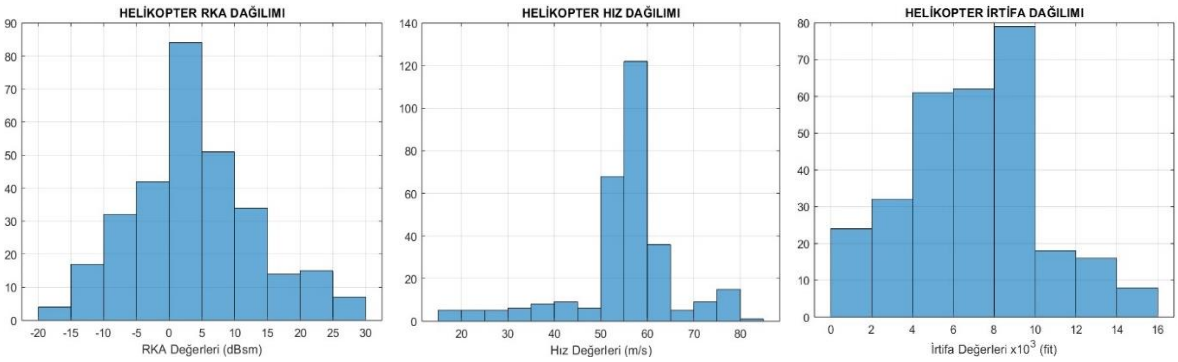
$$Hız = 2\pi r \ \frac{RPM}{60} \tag{3.3}$$

Son olarak ele alacağımız özellik olan irtifa değeri ise hedefin yerden olan yüksekliğini ifade etmektedir.

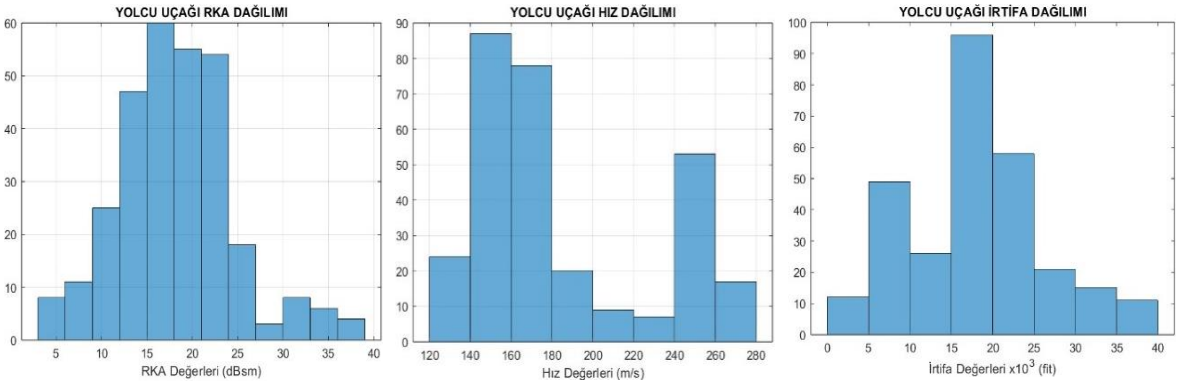
Uygulamalarda kullanılması amacıyla derlenen verilerin dağılımına ilişkin gösterim, Şekil 3.1-7 olarak histogram grafiklerinde sunulmuştur.



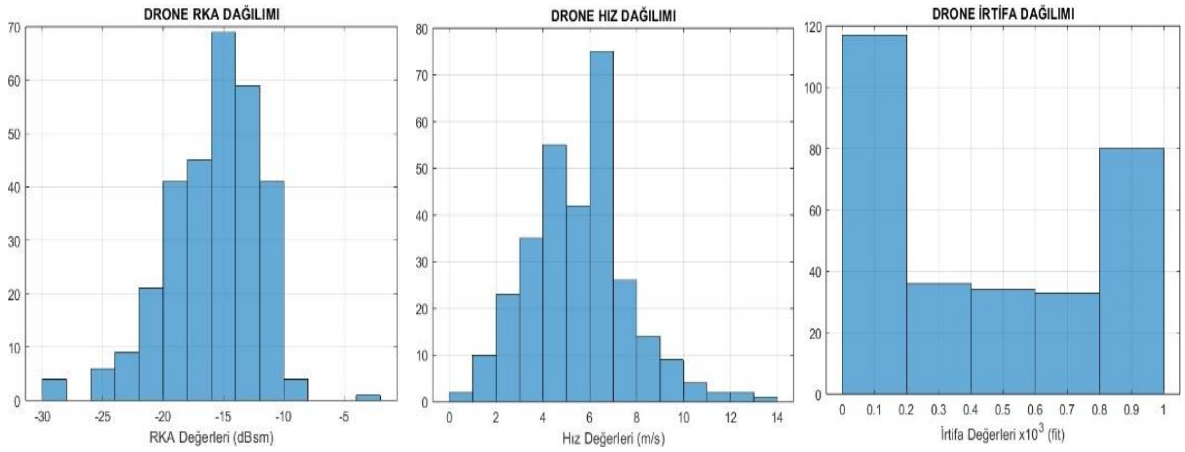
Şekil 3.1. Kuş hedefine ait veri değerleri



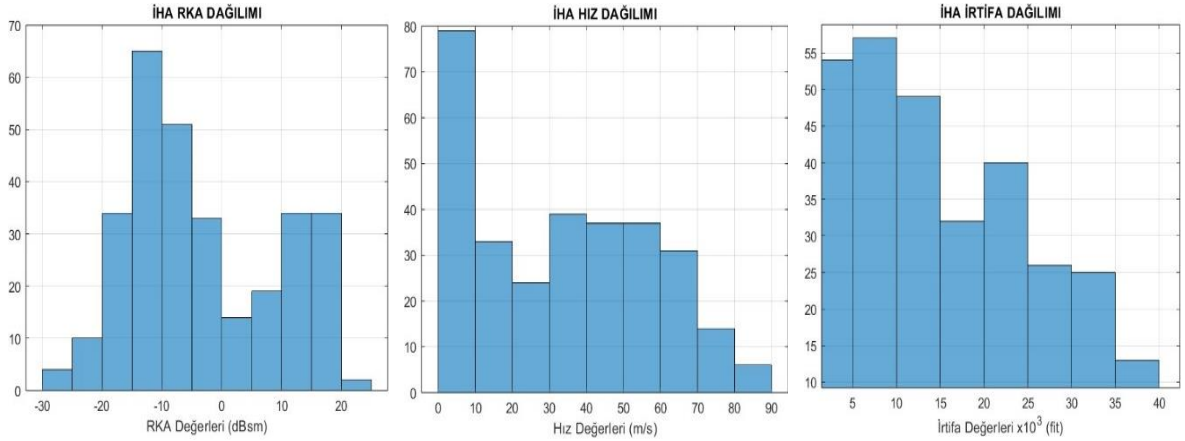
Şekil 3.2. Helikopter hedefine ait veri değerleri



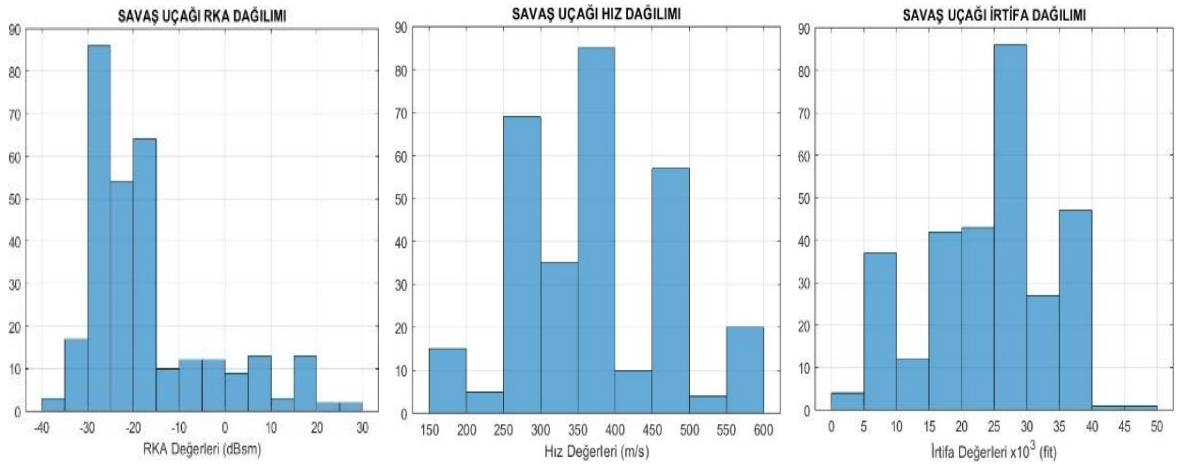
Şekil 3.3. Yolcu uçağı hedefine ait veri değerleri



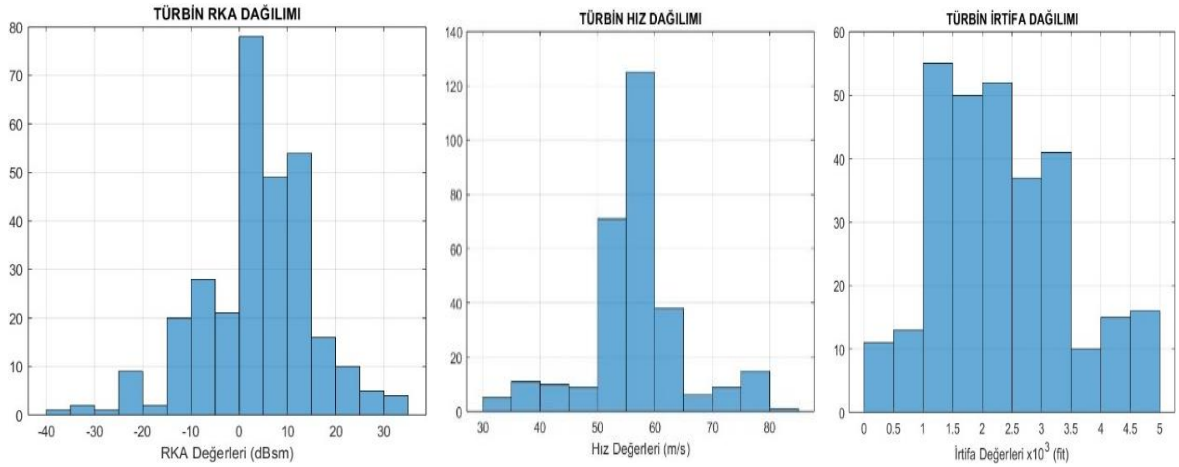
Şekil 3.4. Dron hedefine ait veri değerleri



Şekil 3.5. İHA hedefine ait veri değerleri

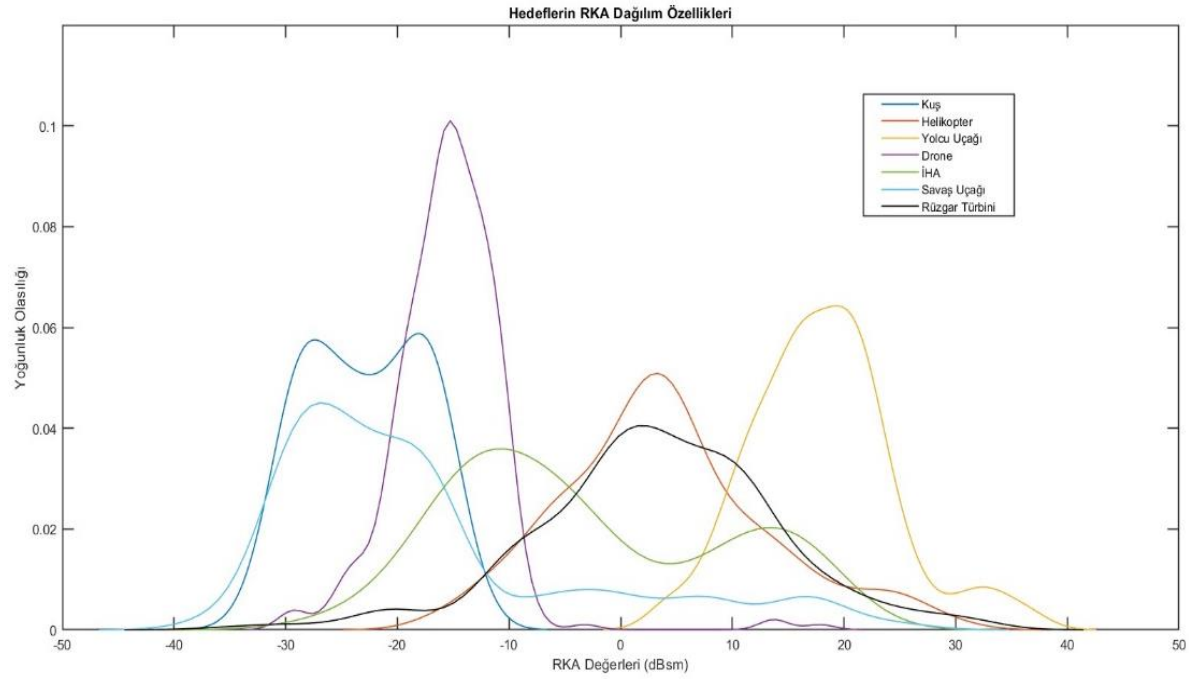


Şekil 3.6. Savaş uçağı hedefine ait veri değerleri



Şekil 3.7. Rüzgâr türbini hedefine ait veri değerleri

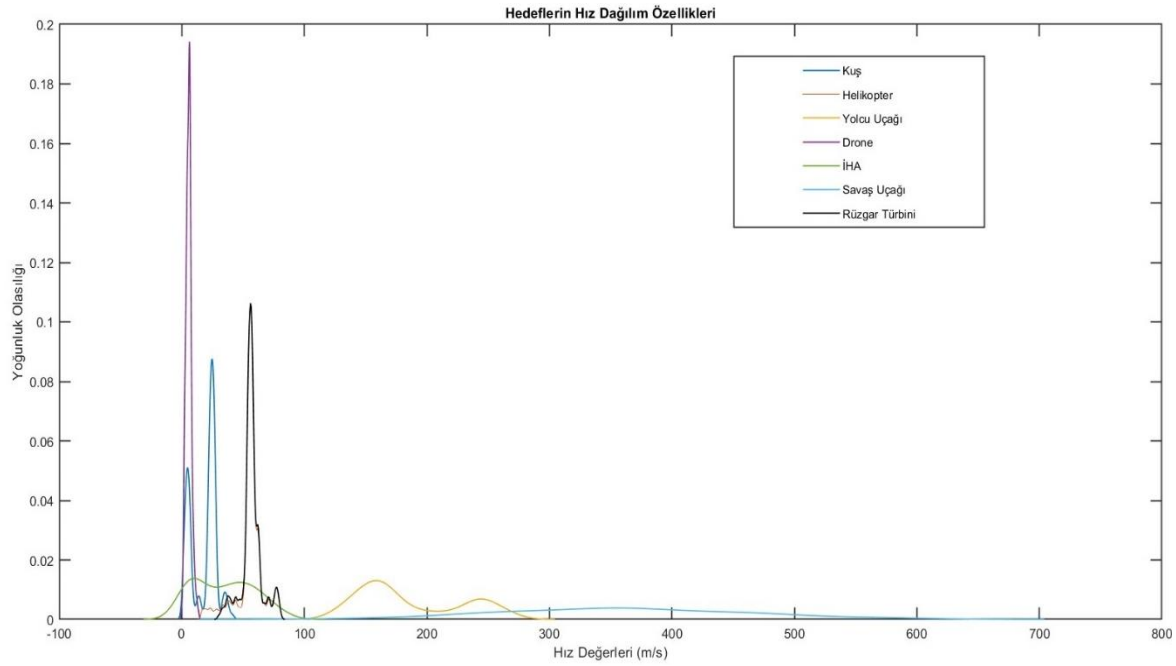
Bu çalışma için veri toplamak amacıyla açık bilgi kaynaklarından (makaleler, cihaz katalogları, vb.), seçilen hedeflere ait tasarım ve kurulum gibi kriterlerden ve test alanlarından elde edilmiş çalışmalardan yararlanılmıştır [22-42]. Askeri ve ticari anlamda gizlilik değerleri bulunan bazı veri setlerinin açık bilgi kaynaklarında bulunmaması nedeniyle bazı veriler yapay olarak oluşturulmuştur.



Şekil 3.8. Tüm hedef sınıflarına ait RKA değerleri

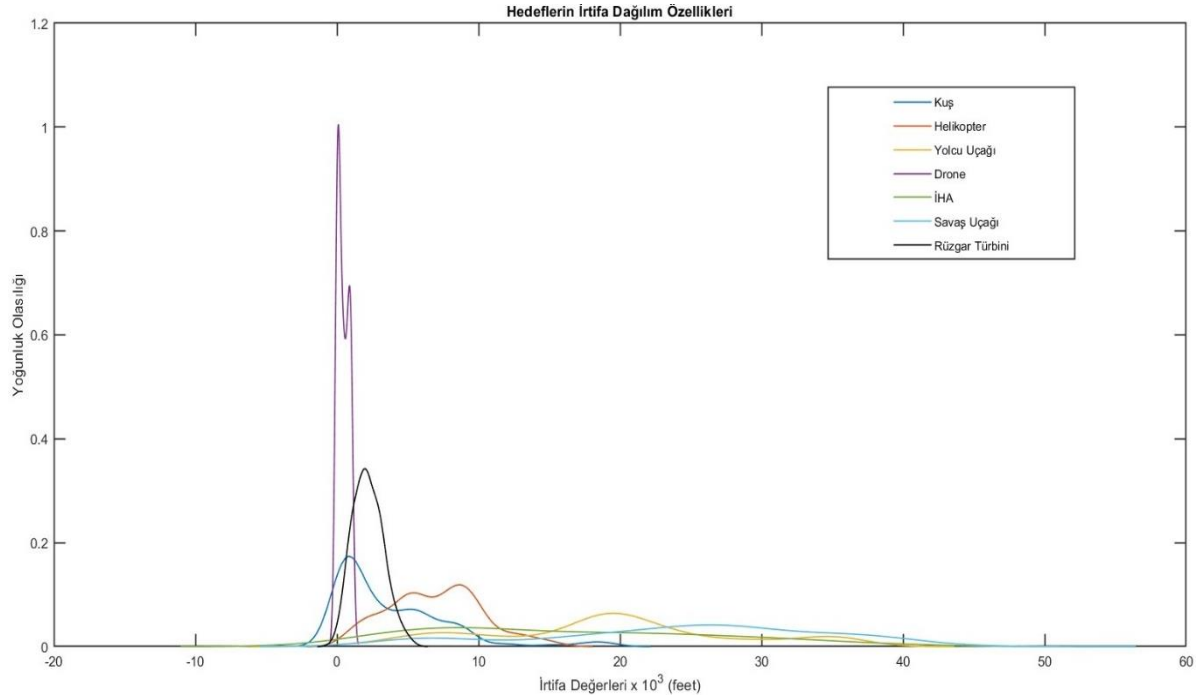
Hedeflere ait RKA değerlerinin Şekil 3.8'deki yoğunluk tahmini (ksdensity) ile ayırt edici bir özellik olabileceği görülmektedir. Örneğin, dronlar -15 dBsm civarında daha büyük yoğunluk gösterirken, yolcu uçakları +18 dBsm değerinde göstermektedir. Bununla birlikte,

helikopter, rüzgâr türbini ve İHA hedefleri daha geniş bir alana RKA değerlerinin yayıldığı görülmektedir. Öte yandan, savaş uçakları ve kuşlar -20 RKA dBsm civarında daha büyük yoğunluk oranlarına sahiptir.



Şekil 3.9. Tüm hedef sınıflarına ait hız değerleri

Hız verileri açısından (Şekil 3.9), en yüksek olasılık 5 m/s ortalama ile dronlar için mevcuttur. Yolcu uçakları ve savaş uçakları ise en yüksek hızlara sahiptir. Rüzgâr türbini kanatlarının ortalama hız değeri 57 m/s'dir ve helikopterler ve İHA'lar bu açıdan türbin kanatlarına kıyasla biraz daha düşük hıza sahiptir. Şekil 3.9'a göre en yüksek olasılık yoğunluğu dronlar için elde edilmiştir.



Şekil 3.10. Tüm hedef sınıflarına ait irtifa değerleri

Şekil 3.10'a göre hedeflere ait irtifa değerleri helikopter, İHA, yolcu uçakları, savaş uçakları için önemli ölçüde artmaktadır. Tüm hedef verileri 100 feet'ten 45.000 feet'e kadar değişen irtifa değerlerine sahiptir. Ayrıca diğer hedefler, dronlardan farklı olarak daha düşük olasılık yoğunluklarına sahiptir.

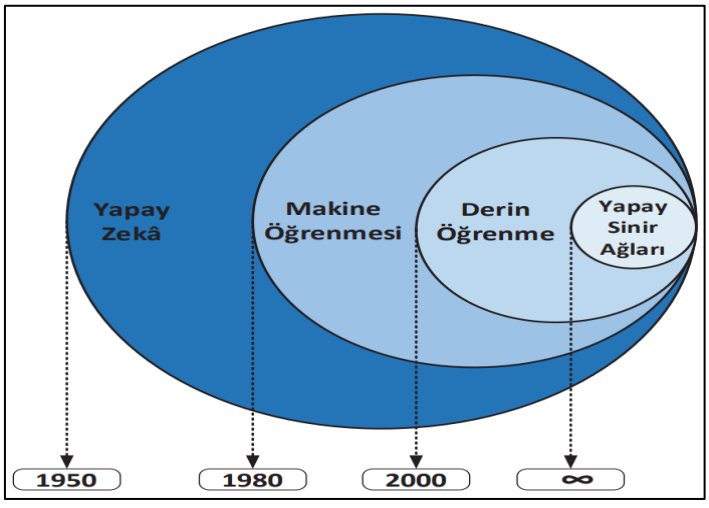
Şekil 3.8-10 ile yoğunluk kestirim yöntemiyle verilerin görselleştirilmesinin ardından, hedefler için toplanan verilerin ortalama ve standart sapma değerleri Çizelge 3.1'de sunulmaktadır. Bu şekilde, veri setinin merkez noktaları ve veri seti değerlerinin ortalamadan ne kadar uzak olduğu sayısal olarak görülmektedir.

Çizelge 3.1. Verilerin ortalama ve standart sapma değerleri

	Ortalama RKA (dBsm)	Std Sapma RKA (dBsm)	Ortalama Hız (m/s)	Std Sapma Hız (m/s)	Ortalama İrtifa x10^3 (feet)	Std Sapma İrtifa x10^3 (feet)
KUŞ	-22,7960	5,1760	18,3356	10,3354	3,7726	4,0136
HELİKOPTER	3,8830	9,2923	55,3476	11,1381	6,9481	3,2592
YOLCU UÇAĞI	18,0296	6,3628	186,1881	42,2901	19,0946	8,7471
DRONE	-15,3871	5,0309	5,6097	2,2019	0,4431	0,3736
İHA	-3,1605	12,1510	34,2820	23,7914	16,4008	10,1719
SAVAŞ UÇAĞI	-17,3408	13,8377	361,520	96,6341	24,1656	9,9951
RÜZGAR TÜRBİNİ	3,2614	10,7209	56,8476	8,4312	2,1915	1,0248

4. YAPAY ZEKA SINIFLANDIRMA YAKLAŞIMLARI

Özellikle son on yıldır olmak üzere giderek daha yoğun bir şekilde konuşulup tartışılan yapay zeka konusuna bakıldığında, farklı alanlar için birçok tanımlamaların olduğu görülmektedir. Bu tanımlamaların arasında yapay zekayı, bilgisayarların ve makinelerin insan zekâsını taklit ederek öğrenme, problem çözme, karar verme gibi görevleri yerine getirme kabiliyetine sahip olduğu bir bilim dalı olarak kabul edebiliriz. Bu kavram ilk kez 1956 yılında Dartmouth Konferansı'nda John McCarthy tarafından önerilmiştir ve o zamandan beri sürekli gelişerek çeşitli alt dallara ayrılmıştır [43]. Yapay zekâ ile hedeflenen amaç, karmaşık sorunların ele alınarak daha hızlı ve verimli çözülebilmesi için insan zekasını taklit edecek bir yapının oluşturulmasıdır. Bu yapıyla birlikte insan gibi düşünen ve bu şekilde oluşturulan düşünce mekanizmasını çok daha hızlı yürüten bir teknoloji gelişmiştir ve gelişmektedir. Yapay zekâ sayesinde hızlı çözümlerin bulunması, yöntem ve yaklaşımların yüksek mantık çerçevesinde modellenmesi ile sağlanmaktadır. Yapay zekâyı oluşturan unsurların bir kısmı makine öğrenmesi ve derin öğrenmedir. Yapay zekâ, makine öğrenimi gerektirmektedir. Derin öğrenme, makine öğrenmesinin bir alt kümesidir. Yapay zekânın tarihsel kronolojisi Şekil 4.1'de gösterilmiştir.



Şekil 4.1. Yapay zeka, makine öğrenmesi ve derin öğrenme kronolojisi [44]

4.1. Yapay Zeka'nın Temel Bileşenleri

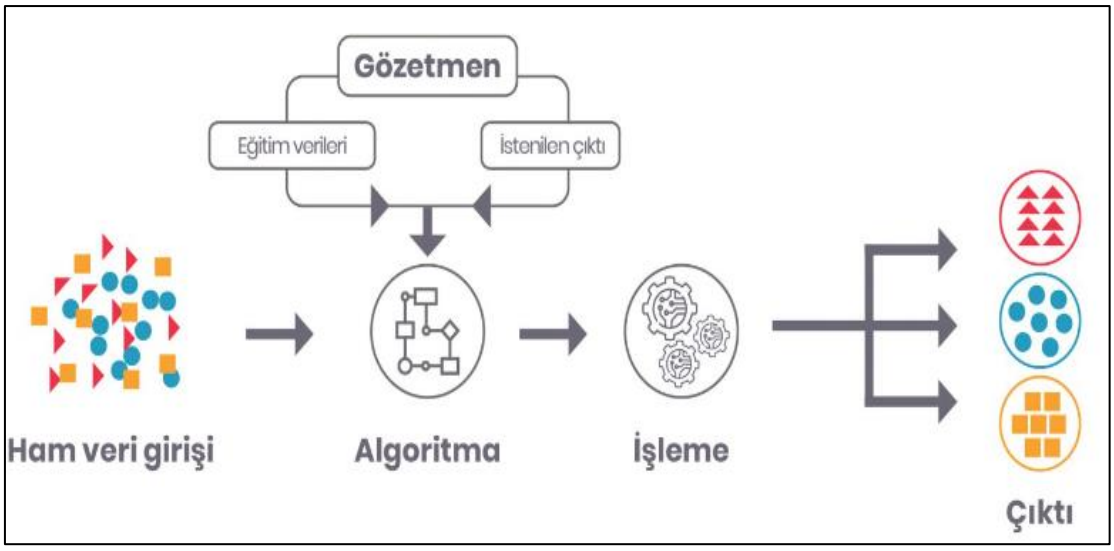
4.1.1. Makine öğrenmesi

Yapay zekanın bir alt dalı olan Makine Öğrenmesi, bilgisayarların verilere dayanarak öğrenme yeteneğine sahip olduğu bir teknolojiyi ifade eder. Bu alanda öncülerden biri olan

Arthur Samuel, makine öğrenmesini "verilerden öğrenebilen bilgisayar sistemleri" olarak tanımlamıştır [45]. Sistemler, algoritmalar kullanarak verileri analiz eder ve bu analizlere dayanarak gelecekteki kararlarını iyileştirir.

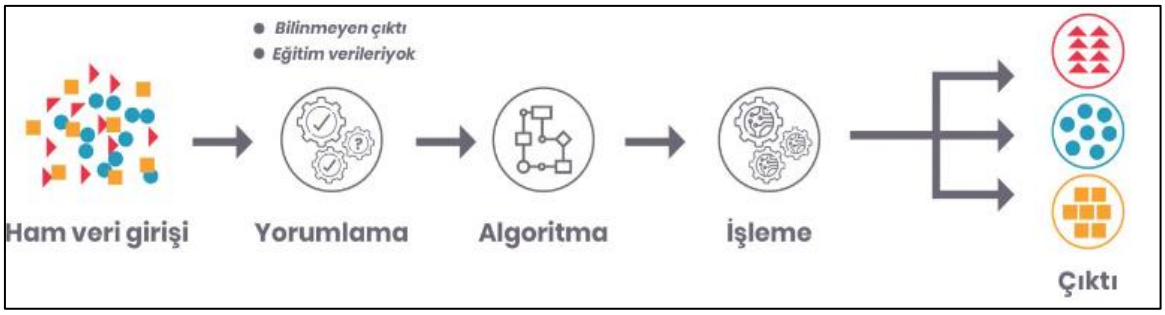
Farklı uygulamalar için tasarlanabilen Makine Öğrenimi algoritmaları, genel olarak denetimli (Supervised Algorithms) ve denetimsiz (Unsupervised Algorithms) olarak iki kategoriye ayrılıp tanımlanabilir.

Şekil 4.2'de işlem basamakları sunulan denetimli Makine Öğrenimi algoritma ile etiketli veri kümesi kullanarak öğrenme yapılmaktadır. İşlemler aşamasında, modelin tasarlanmasına ilişkin sürecin yönetimi denetim mekanizması ile yürütülmektedir. Model tasarlanırken, eğitim verileri etiketlenerek algoritma yönteminin ve çerçevesinin netleşmesi sağlanır. Etiketli veriler (geçmiş öğretiler) referans alınarak test verilerinin (gelecekteki olayların) tahmin edilmesi gerçekleştirilir. Algoritma çıktısındaki tahmini değer ve gerçek değer karşılaştırılmasındaki hataların analiz edilmesiyle modelde değişiklik yapılabilir.



Şekil 4.2. Denetimli makine öğrenimi [46]

Şekil 4.3'te akış şeması sunulan denetimsiz Makine Öğrenimi yönteminde, verilerin filtrelenmesinde denetimsiz algoritmalar kullanılır. Eğitim verilerine ait bir etiketlendirmenin veya verilerin bir kategoriye ayrılmasının olmadığı koşullarda kullanılır. Tasarlanan modelin çıktıyı doğru tahmin edemediği durumlarda, etiketlenmemiş verilerin analiz edilme süreci tekrarlanarak verilerden çıktı elde edilme sürecine devam edilir.



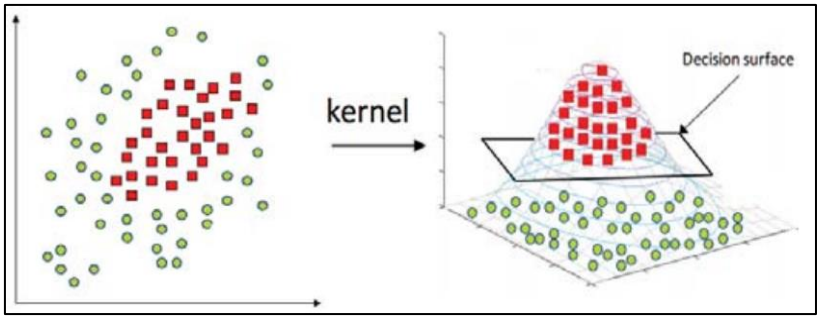
Şekil 4.3. Denetimsiz makine öğrenimi [46]

Bahsedilen iki farklı Makine Öğrenme algoritmalarının yanı sıra Takviyeli Öğrenme, Regresyon, kümeleme gibi yapılarda mevcuttur. Makine öğrenmesi, 1950'lerden bu yana gelişim göstermeye devam etmektedir. Alan Turing'in 1950'de yazdığı "Computing Machinery and Intelligence" adlı makalesi, makine öğrenmesinin ilk teorik temellerini atmıştır [47]. 2010'lu yılların başında, büyük veri ve hesaplama gücündeki artışla birlikte makine öğrenmesi daha geniş çapta uygulanmaya başladı.

4.1.2. Makine öğrenmesinin temel modelleri

Destek vektör makineleri (SVM)

Desen sınıflandırması için kullanılan çekirdek tabanlı yöntemler grubunun bir parçasıdır. SVM, temel olarak doğrusal olarak ayrılabilir verileri sınıflandıran doğrusal bir sınıflandırıcıdır, ancak özellik vektörleri doğrusal olarak ayrılabilir olmayabilir. Bu sorunun üstesinden gelmek için çekirdek hilesi kullanılır [48]. SVM, başlangıçta düşük boyutlu veri kümelerini doğrusal, polinom, radyal baz, sigmoid gibi farklı çekirdek fonksiyonlarına sahip yüksek boyutlu veri kümelerine dönüştürerek doğrusal olmayan ayrılabilir verilerle çalışabilir. Böylelikle, veri kümeleri yüksek boyutlara dönüştürülerek doğrusal olarak sınıflandırma mümkün olmaktadır.



Şekil 4.4. Kernel haritalama işlevi

Şekil 4.4'de, doğrusal olarak ayrılamayan verilerin çekirdek fonksiyonlarıyla doğrusal olarak ayrılabilir verilere dönüştürülme süreci görselleştirilmiştir [49]. Optimum hiper çizgisi (karar sınırları), boyut değişikliklerinden sonra uygulanan SVM algoritmaları ile belirlenir ve hiper çizgisi iki sınıf arasındaki karar aralıkları oluşturulur.

Karar ağaçları (Decision trees)

Veriyi dallara ayırarak kararların alınmasını sağlayan bir modeldir. Hiyerarşik yapı üzerinden sınıflandırma yapılır [50]. Gözlemlerden elde edilen sonuçları işlemeye geçmek için bir karar ağacından yararlanmaktır. Karar ağaçlarında, yukarıdan aşağıya bir yaklaşım benimsenmektedir.

KNN (K-en yakın komşular)

KNN, bazı veri kümelerinde en yakın K veri noktasını bulmak için eğitim veri kümelerini kullanan bir algoritmadır. Bu algoritmanın temel prensibi, bir veri noktasını, kendisine en yakın ‘k’ komşunun sınıfına göre sınıflandırmaktır. Yeni bir veri noktası sınıflandırılacağı zaman, eğitim veri setindeki tüm noktalarla olan mesafesi hesaplanır. Hesaplanan mesafelere göre yeni veri noktasına en yakın ‘k’ adet komşu belirlenir. Seçilen ‘k’ komşunun hangi sınıfa ait olduğuna bakılır. Çoğunlukla hangi sınıf varsa, yeni veri noktası da o sınıfa atanır. Modelin performansı seçilen ‘k’ sayısı ile doğrudan ilişkilidir. Küçük olması aşırı öğrenmeye, büyük olması modelin doğruluğunu azaltabilir. [51]

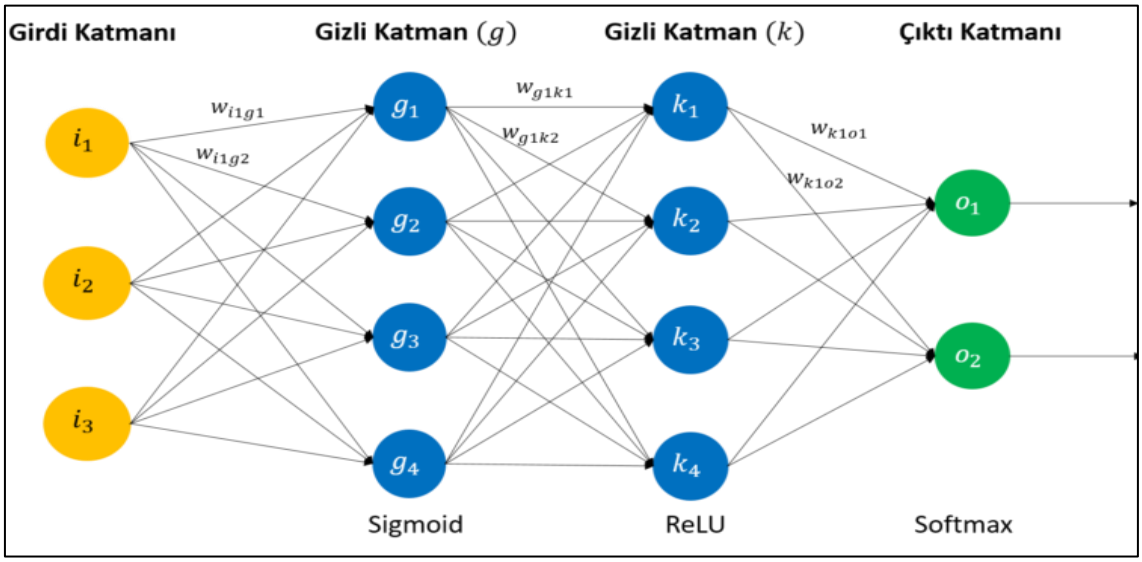
Topluluk (Ensemble) algoritması

Ensemble (Topluluk) yöntemleri, birden fazla makine öğrenmesi modelini birleştirerek tek bir güçlü model oluşturmayı amaçlayan yaklaşımlardır. Bu yöntemler, farklı modellerin zayıf yönlerini telafi edip, genelde daha iyi genel doğruluk ve genellenebilirlik sağlarlar. En yaygın ensemble yöntemleri Bagging, Boosting ve Stacking gibi tekniklerdir.

4.1.3. Derin öğrenme

Makine öğrenmesinin bir alt dalı olan Derin Öğrenme yapay sinir ağlarının daha derin yapılı hallerine verilen isimdir. Şekil 4.5’de genel şeması verilen yapay sinir ağları, her birinde özel hesaplamaların yapıldığı nöronlar ve bunların gruplaştığı girdi katmanı, gizli katman ve çıktı katmanı olarak tanımlanan yapılardan oluşur. Bir sinir ağında katmanlar içerisinde yer alan bir nöronu diğer katmandaki nörona bağlayan bağlantılar ve bu bağlantıların sayısal

değerleri (ağırlık) kullanılarak nöronun girdi ve çıkış etkileri hesaplanır. Hesaplama sonucunda modelin omurgası tasarlanmış olur.



Şekil 4.5. Yapay sinir ağının genel yapısı [52]

Özellikle büyük veri kümelerinde etkili sonuçlar elde eden derin öğrenme teknolojisi, görüntü ve ses tanıma gibi uygulamalarda kullanılmaktadır. Kendi oluşturduğu öznitelik verilerini kullanarak geliştirdiği matematik modelleri ile öğrenme sürecini belirler. Derin öğrenmenin kökleri, 1940'lara kadar uzanmaktadır. 1943'te McCulloch ve Pitts, sinir hücrelerinin işlevini matematiksel olarak modelleyen ilk yapay sinir ağını geliştirdi. Ancak, derin öğrenme olarak bilinen modern uygulamaların yükselişi 2006 yılında Geoffrey Hinton'un öncülüğünde gerçekleşti. Hinton, derin sinir ağlarının eğitimine yönelik yeni yöntemler önerdi ve bu alanın canlanmasına yardımcı oldu [53].

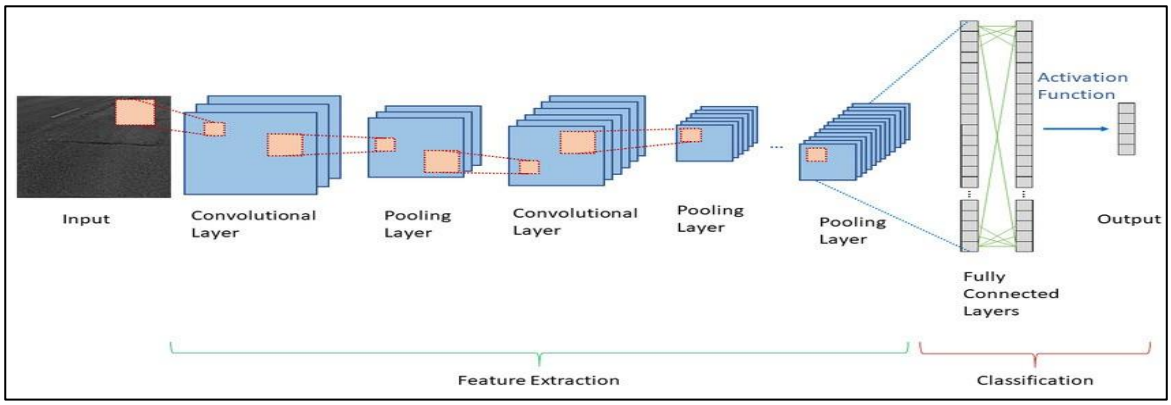
4.1.4. Derin öğrenmenin temel modelleri

Yapay sinir ağları (ANN)

Biyolojik sinir ağlarından ilham alınarak geliştirilen ve derin öğrenme modellerinin temel yapı taşlarındandır. Birden fazla katman içeren sinir ağlarıyla daha karmaşık veri yapıları üzerinde öğrenme yeteneğine sahiptir. Gizli katmanlar arasında ReLU (Rectified Linear Unit), Sigmoid ve Tanh gibi aktivasyon fonksiyonları kullanırken, Çıktı katmanında genellikle softmax veya sigmoid aktivasyon fonksiyonları sayesinde her sınıf için olasılıklar hesaplanması için temel oluşturur.

Evrişimli sinir ağları (CNN)

Özellikle, görüntülerdeki uzamsal yapıları tanıyarak sınıflandırma yapar. Derin Öğrenmede en çok kullanılan modellemedir.

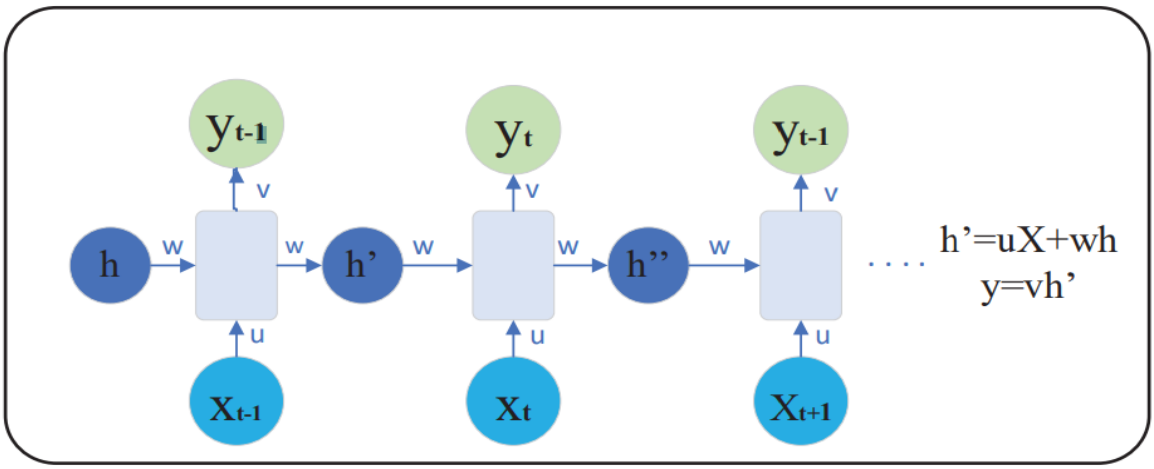


Şekil 4.6. Evrişimli sinir ağı katmanları [54]

Şekil 4.6'da genel tasarım mimarisi olarak sıralı işlem basamakları görülen Evrişimli Sinir Ağları; konvolüsyon (convolution), düzleştirme (rectified linear unit), havuzlama (pooling) ve tam bağlantı (fully connection) katmanlarından oluşur.

Tekrarlayan sinir ağları (RNN)

Zaman serisi ve ardışık veri analizi gibi görevler için kullanılan Tekrarlayan Sinir Ağları, önceki çıkışlarını bir sonraki giriş olarak kullanarak bir döngü oluşturan ve bu döngü esnasında bilgilerin sıralı olarak kullanılmasına özen gösteren öğrenme ve tahmin temelli bir yapay sinir ağı modelidir. Bu ağ yapısı ile bilgilerin kalıcı olarak döngülerde kalmasına ve gerektiği zaman kullanılmasına izin verilmektedir. [55]



Şekil 4.7. Tekrarlayan sinir ağları işlem akışı [35]

Giriş verileri birbirlerinden bağımsız olmasına rağmen her verinin çıktısı önceki verinin hesaplamalarına bağlı olarak model geliştirilir. RNN işlem akışının gösterildiği Şekil 4.7'de, herhangi bir t zamanında, h aktivasyonu, x girdiyi ve y çıktıyı temsil etmekte olup 'h' önceki çıkış aynı zamanda bir sonraki giriş olarak modelin oluşmasını sağlamaktadır.

4.1.5. Makine öğrenmesi ile derin öğrenmenin benzerlikleri ve farkları

Benzerlikler

- Her iki alan da verilerden öğrenmeye dayanır.
- Denetimli (Supervised Algorithms) ve denetimsiz (Unsupervised Algorithms) öğrenme yöntemleri her iki yapı için de kullanılabilir.
- Genelde yapay sinir ağları üzerine kuruludur ve büyük veri kümelerinden faydalanır.

Farklar

- Veri Miktarı: Derin öğrenme, genellikle büyük veri kümelerine ihtiyaç duyar, makine öğrenmesi ise daha küçük veri setleriyle çalışabilir.
- Hesaplama Gücü: Derin öğrenme modelleri, yüksek hesaplama gücü gerektirirken, makine öğrenmesi daha basit algoritmalarla da işleyebilir.
- Model Karmaşıklığı: Derin öğrenme çok katmanlı yapılarla karmaşık problemlerde daha iyi performans gösterirken, makine öğrenmesi daha yüzeysel ve basit çözümler sunabilir.

Genel olarak, makine öğrenmesi daha basit ve hızlı çözümler sunarken, derin öğrenme daha karmaşık problemlerde güçlü sonuçlar ortaya koyar. Bu iki öğrenme yaklaşımı, veri analitiği ve yapay zekâ uygulamaları için büyük önem taşımaktadır.

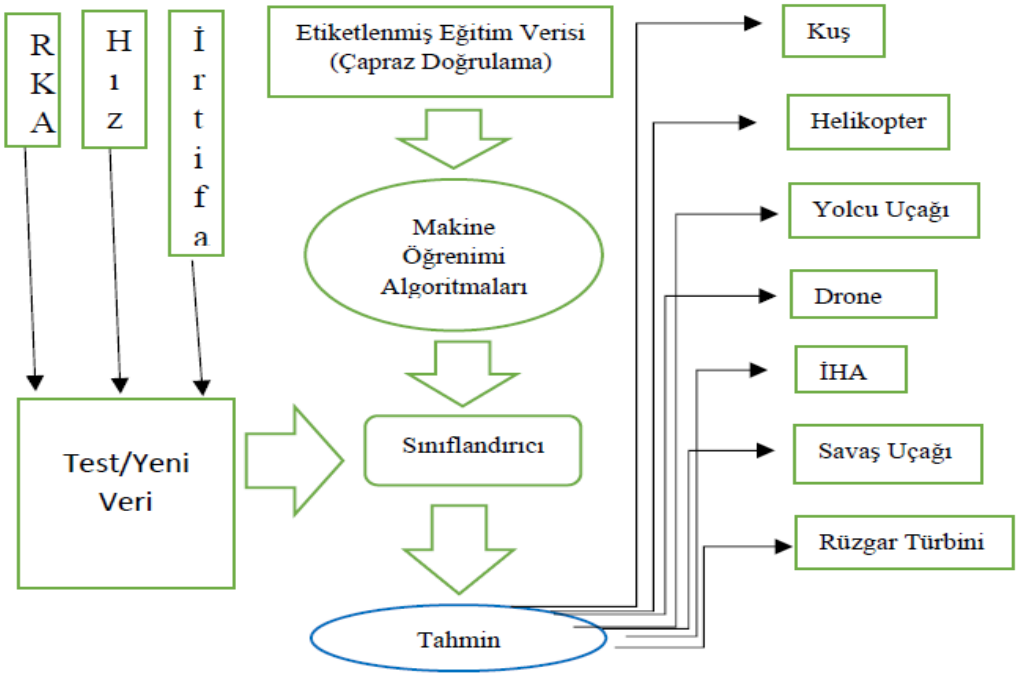
5. YAPAY ZEKA SINIFLANDIRMA UYGULAMALARI

Tez kapsamında, hem makine öğrenmesi hem derin öğrenme yöntemi ile uygulamalar gerçekleştirerek sonuçların analiz edilebilmesi amacıyla 3 uçuş verisinden (RKA, hız, irtifa) oluşacak şekilde 7 farklı hedefe ait 300'er veri hazırlanmıştır. Uygulamalar MATLAB R2023b versiyonu üzerinde yapılmıştır.

İlk etapta veriler % 85 eğitim ve %15 test olacak şekilde iki farklı gruba ayrılmıştır. Başka bir ifade ile toplam 300 adet uçuş verisinden 255 örnek eğitim kümesi için, 45 örnek ise test kümesi için seçilmiştir. Eğitim veri kümesi modeli oluşturmak için kullanılan örnekleri ifade ederken, test veri kümesi modelin performansını değerlendirmek için kullanılan örneklerdir.

5.1. Makine Öğrenimine ait Uygulamalar

Makine Öğrenimi'nde sınıflandırma yapılması için MATLAB uygulamalarında Sınıflandırma Öğreticisi (Classification Learner) tercih edilmiştir. Uygulama içinde çeşitli sınıflandırma algoritmaları bulunmakta olup çeşitli algoritmaların denenmesi, karşılaştırılması ve performans değerlendirmesinin yapılabilmesi de mümkün olmaktadır.



Şekil 5.1. Sınıflandırma öğreticisine ait işlem basamakları

Şekil 5.1'de işlem basamakları sunulan ve denetimli bir öğrenme olarak kabul edilen ssınıflandırma algoritmaları sayesinde test verisi (yeni bir öğe) ile tanımlanan hedefin

önceden tanımlanan (etiketlenen) 7 adet sınıftan hangisine ait olma olasılığı hesaplanır. Bu sayede bir hedef sınıflandırma tahmini gerçekleştirilmiş olur. Bu yönteme ilişkin, gelecekteki çıktıları tahmin etmek üzere etiketi bilinen giriş ve çıkış verileri üzerinde bir modeli eğiten bir iş akışı diyagramı gösterilmektedir. Yeni/test verilerine yanıt (sınıf) için makul tahminler üretmek üzere bir model eğitir. Bu çalışmada kullanılan denetimli öğrenme tekniğinin seçilmesinin altında yatan gerekçe, tahmin etmeye çalıştığımız çıktı (etiketli) verilerinin önceden bilinmesidir.

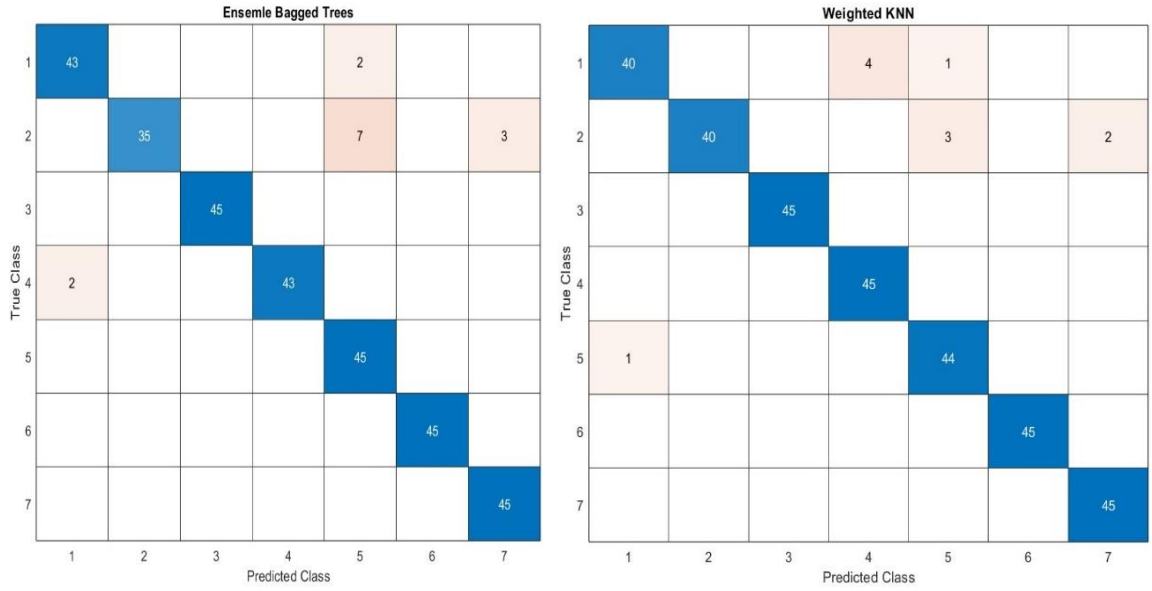
Her bir uçuş değeri için 255 veri içeren eğitim kümesinin işlendiği makine öğrenmesi algoritmalarında 5 katlı çapraz doğrulama yapılmıştır. Böylelikle, eğitim verileri beş alt kümeye bölünmüş, her biri bir alt kümeyi doğrulama kümesi, diğer dört alt kümeyi de eğitim kümesi olarak seçilmiştir. Tüm eğitilmiş modeller, tüm bu işlemlerin beşer kez tekrarlanmasıyla elde edilmiştir. Doğrulama prosedürlerinin tamamlanmasını müteakip, tüm modellere 45 örnek içeren bir test seti uygulanmıştır. Daha sonra, bilinen etiket değerlerine sahip test setindeki modelin performans kriterleri değerlendirilmiştir. Genel olarak bakıldığında MATLAB uygulamalarında, sınıflandırma amacıyla yaygın olarak kullanılan Vektör Makineleri, Karar Ağaçları, K-En Yakın Komşular, Naive Bayes, Ayırıcı, Lojistik Regresyon ve Yapay Sinir Ağları gibi makine öğrenmesine ait çeşitli algoritmalar bulunmaktadır. Kullanılan veri kümelerine bağlı olarak, bu algoritma modelleri farklı uygulama alanlarında başarı, hız ve yöntem kolaylığı açısından birbirlerine göre avantajlara veya dezavantajlara sahip olabilir. Bu çalışmada, uygulama yapılan 5 farklı algoritma modelinin doğrulama ve test aşamaları neticesinde elde edilen doğruluk oranları Çizelge 5.1'de sunulmuştur.

Çizelge 5.1. Sınıflandırma öğreticisine ait doğruluk oranları

Algoritma Modeli	Doğruluk Oranı (Doğrulama)	Doğruluk Oranı (Test)
Kernel Naive Bayes	% 88,6	% 92,1
Ensemble Bagged Trees	% 90,4	% 95,6
Medium Neural Network	% 88,7	% 92,7
Weighted KNN	% 88,2	% 96,5
Cubic SVM	% 89,2	% 94,6

Çizelge 5.1'de görülebileceği gibi, daha iyi sonuçlar elde etmek için, doğruluk oranlarındaki başarılar göz önünde bulundurulduğunda en makul algoritmaların Topluluk Torbalama Ağaçları (Ensemble Bagged Trees) ve Ağırlaştırılmış En Yakın Komşular (Weighted KNN) olduğu belirlenmiştir. Makine Öğrenmesi terminolojisinde, sınıflandırma modelinin

performansını ölçmek için karışıklık matrisi kullanılır. Sınıflandırma modeline daha önce görmemiş olduğu ve gerçek etiket değerlerinin bilindiği test kümesinin girdi olarak verilmesiyle görselleştirilen karışıklık matrisi ile doğru ve yanlış tahmin edilen örnekler arasındaki oran belirlenir. Yukarıda tavsiye edilen sınıflandırma modelleri kullanılarak elde edilen başarı oranları Şekil 5.2'de sunulmuştur.



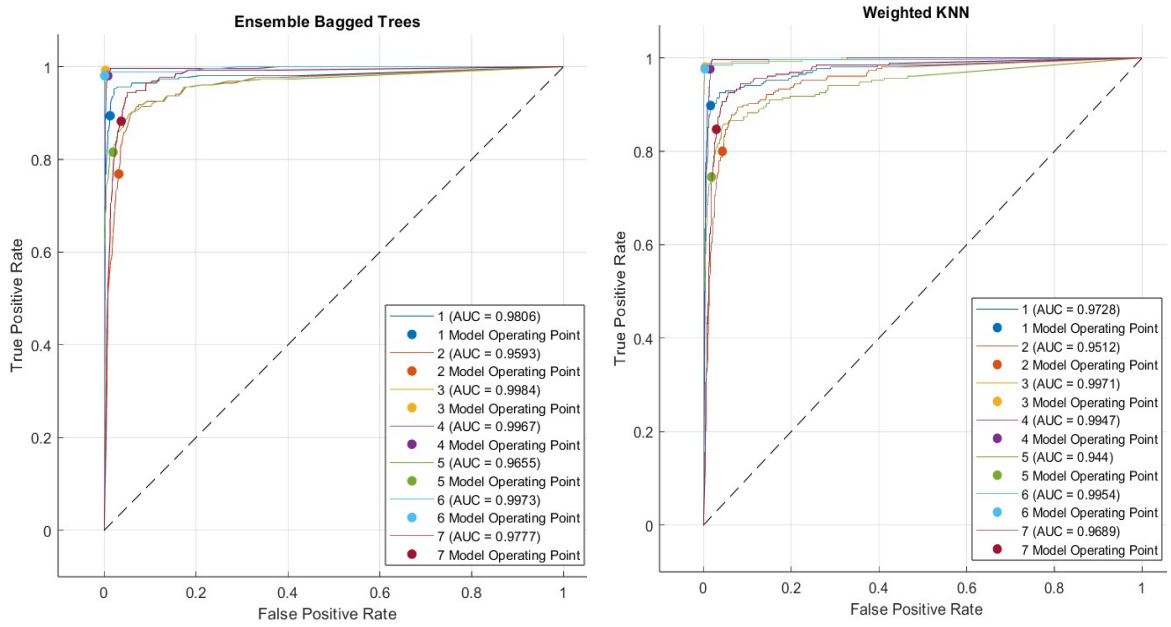
Şekil 5.2. Topluluk torbalama ağaçları ve ağırlaştırılmış en yakın komşular modellerinin karışıklık matrisleri

"1" rakamıyla etiketlenen hedef sınıfı kuş ve "2" ile "7" arasındaki diğer rakamlar ise sırasıyla helikopter, yolcu uçağı, dron, İHA, savaş uçağı ve rüzgâr türbinini olarak tanımlanmıştır.

Bu diyagramlarda her sütun öngörülen sınıf etiketine, her satır ise gerçek sınıf etiketine karşılık gelmektedir. Bu şekilde test örneklerinin kaç tane doğru/yanlış olarak sınıflandırıldığı açıkça görülmektedir. Örneğin, modelin performansını görselleştirmeye ve özetlemeye olanak tanıyan Ensemble Bagged Trees Karışıklık Matrisi ile 45 test verisinden 35'inin '2' etiket numarasıyla belirttiğimiz helikopter sınıfına ait olarak doğru tahmin edildiğini ve %77,8'lık bir başarı oranına ulaştığını, diğer % 22,2'lik yanlış tahminlerin ise '5' ve '7' etiket numaraları ile sırasıyla İHA ve rüzgar türbini olarak tahmin edildiğini göstermektedir. Tüm sınıflar için % 95,6'lık bir başarı seviyesi elde edilmiştir. Weighted KNN modeline ait Karışıklık Matrisi verileri incelediğinde, farklı bir örnek olarak, '1' etiket numarasıyla kuş hedefine ait 45 adet test verisinden 40 adedinin doğru tahmin edilmesiyle % 88,9 başarı oranı yakalanmıştır. İki modeli genel olarak karşılaştırdığımızda, birbirlerine

yakın başarı oranları olmasına ragmen Ensemble Bagged Trees modelinin İHA hedef tespitlerinde Weighted KNN modeline göre daha başarılı olduğu görülmektedir.

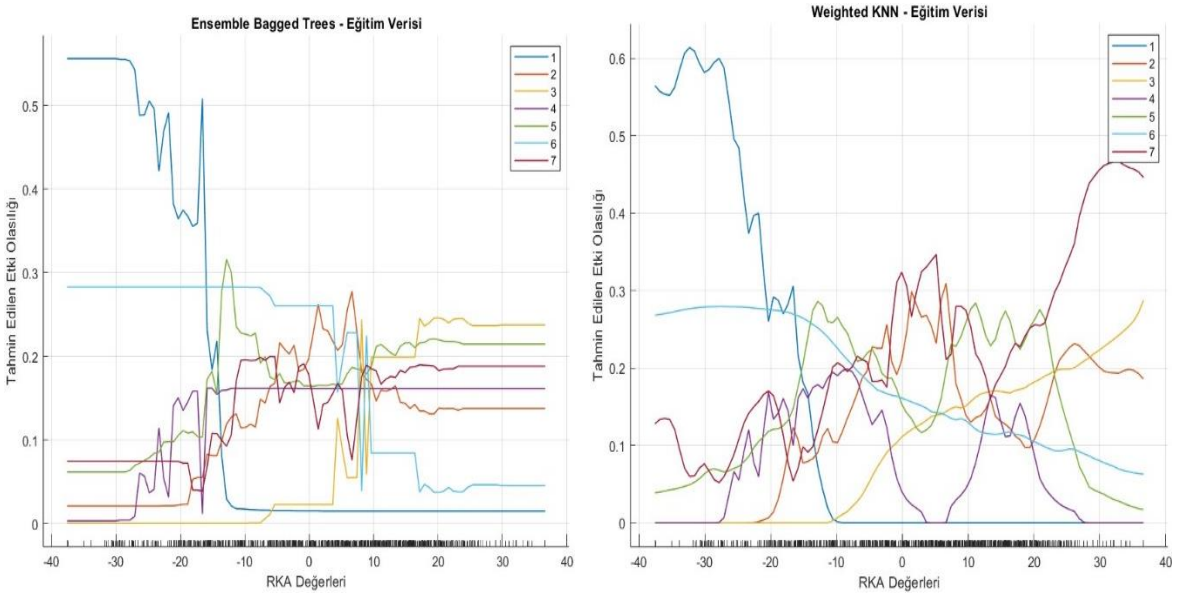
Makine Öğrenimi sınıflandırmalarında modele ait tahminlerin doğruluğu ve etkinliğini değerlendirmeye yardımcı olan karışıklık matrisinin yanı sıra grafiksel gösterimleri ile Alıcı İşletim Karakteristiği (ROC) de diğer bir performans analiz aracıdır. Şekil 5.3'de sunulan grafikte Eğri Altındaki Alan (AUC) değerlerinin bire yakın olduğu durumda modellerin yüksek performans göstereceği ve modellerin ihtiyacı karşılayacak düzeyde eğitildiği değerlendirilmektedir. Bu doğrultuda, her iki modelde de en iyi değer "3" ile etiketli yolcu uçağı sınıfında elde edilmiştir.



Şekil 5.3. Topluluk torbalama ağaçları ve ağırlaştırılmış en yakın komşular modellerinin ROC analizleri

Şekil 5.4, 5.5 ve 5.6'da görülen Kısmi Bağımlılık Grafikleri (PDP-Partial Dependence Plot), Makine Öğrenmesi özelinde incelediğimiz iki model tarafından yapılan tahminlerin hangi özelliklerden nasıl etkilendiğini daha iyi anlamamıza katkı sunar. PDP grafikleri, belirli bir özelliğin (RKA, hız ve irtifa) tahmin sonuçlarına etkisini izole ederek göstermektedir. PDP grafiklerinde x ekseni, analiz edilen bağımsız değişkenin değerlerini temsil eder. Örneğin, hız verisi için x ekseninde hız değerleri sıralanır. Bu eksende görülen veri sıklığı, o özellik değerinin tahmin üzerindeki etkiyi doğrudan etkilemektedir. Y ekseni, modelin tahmin ettiği değerleri (bağımlı değişken) temsil eder. Yani, x eksenindeki değişkenin farklı değerlerinde modelin nasıl bir tahminde bulunabileceğine yönelik değerlendirme olanağı sağlar. Grafiklerdeki çizgi seyirleri, analiz edilen özelliğin tahmin değerine etkisini

görselleştirmektedir. Çizgi yukarı doğru seyrediyorsa, x eksenindeki özelliğin tahmin edilen değeri artırma eğiliminde olduğu anlamına gelmektedir. Eksenlerde görülen veri dağılımı, modelin hangi değer aralıklarında tahmin yaparken daha fazla veriye dayandığını göstermektedir. Bu dağılım, tahminlerin güvenilirliği açısından önemlidir; çok fazla veya çok az veri dağılımı, modelin tahmin doğruluğunu etkileyebilmektedir. PDP grafikleri ayrıca, hangi özelliklerin modele daha fazla katkı sağladığını analiz etmek için kullanılır. Şöyle ki, modelin tahmin çizgisi belirgin bir doğrusal veya doğrusal olmayan ilişki gösteriyorsa, ilgili değişkeninin tahminlerde güçlü bir etkisi olup olmadığı sonucuna varılabilir. Daha düz çizgi seyirleri olan grafikler, modelin o özelliğe karşı daha az duyarlı olduğunu gösterir. Özetle, PDP grafikleri, özelliklerin model tahminleri üzerindeki etkilerini izole ederek gösterdiği için, modelin nasıl tahmin yaptığı konusunda bilgi vermekte, yani modelin tahminlerinin daha iyi anlaşılabilir kılmaktadır.



Şekil 5.4. RKA verisinin PDP grafiği

Şekil 5.4'te yer alan PDP ile incelenen iki Makine Öğrenme modeline ait RKA verisinin sınıflandırma sürecindeki etkisi görselleştirilmiştir. Böylelikle, her iki modelin aynı veri üzerinde nasıl farklı tahminler yaptığını anlamak ve hangi modelin RKA verisiyle daha etkili tahminlerde bulunduğuna ilişkin değerlendirme yapılmasına olanak sağlamaktadır. Daha dik eğimli çizgiler, RKA değişikliklerine karşı modelin daha hassas olduğunu, yatay çizgiler ise duyarsız olduğunu belirtir. Hangi modelin daha iyi performans gösterdiği, RKA verilerinin sınıflandırmada ki önem derecesine göre değişir. Eğer RKA değeri sınıflandırmada kritikse, bu değere daha duyarlı model daha iyi performans gösterir. RKA değerlerinin her iki model

üzerindeki etkileri incelendiğinde, her sınıf için modellerin farklı performans gösterdiği ve belirli sınıflarda bir modelin diğerine kıyasla farklı değerlendirme kriterlerinin olduğu görülmektedir.

RKA değerine yönelik PDP her iki modelin 7 sınıf için gösterdiği performans şu şekildedir:

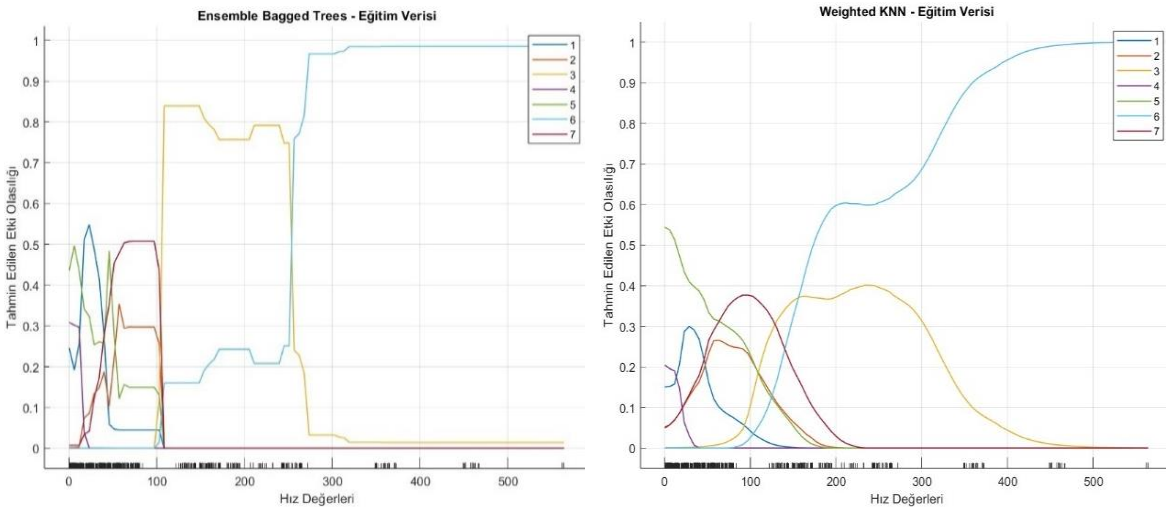
Sınıf 1: Ensemble Bagged Trees, sınıf 1 için RKA değişimlerine karşı daha istikrarlı bir seyir gösterirken, Weighted KNN modelinde sınıf 1'de belirgin dalgalanmalar gözlenmektedir. Bu, Ensemble Bagged Trees'in bu sınıfta daha güvenilir olabileceğini gösterir.

Sınıf 2: KNN modelinde, sınıf 2 için RKA değerlerinde artışa bağlı olarak tahmin değerlerinde belirgin bir artış görülürken, Ensemble Bagged Trees daha düz bir çizgi ile sınıfı ayırt etmektedir. Bu durum, sınıf 2'nin KNN modelinde daha belirgin ayrıştığını göstermektedir.

Sınıf 3: Her iki modelde de sınıf 3'ün RKA değeri arttıkça tahmin değeri yükselmekte ancak Weighted KNN modelinde daha keskin bir eğim vardır. Bu, Weighted KNN modelinin sınıf 3'ü daha duyarlı bir şekilde sınıflandırdığını gösterir.

Sınıf 4: Ensemble Bagged Trees, sınıf 4'te nispeten stabil kalırken Weighted KNN modelinde belirgin bir doğrusal artış vardır. Bu, RKA verisinin sınıf 4 için Weighted KNN'de daha etkili olduğunu düşündürür.

Sınıf 5-7: Son üç sınıf için Ensemble Bagged Trees daha kararlı bir seyir gösterirken, Weighted KNN modelinde bu sınıflar RKA değişimlerine karşı daha duyarlı tepkiler veriyor. Genel olarak, Ensemble Bagged Trees modeli daha düşük varyansla sınıflandırma yaparken, Weighted KNN modelinde RKA değeri sınıflar arasındaki ayrımı daha belirgin hale getiriyor.



Şekil 5.5. Hız verisinin PDP grafiği

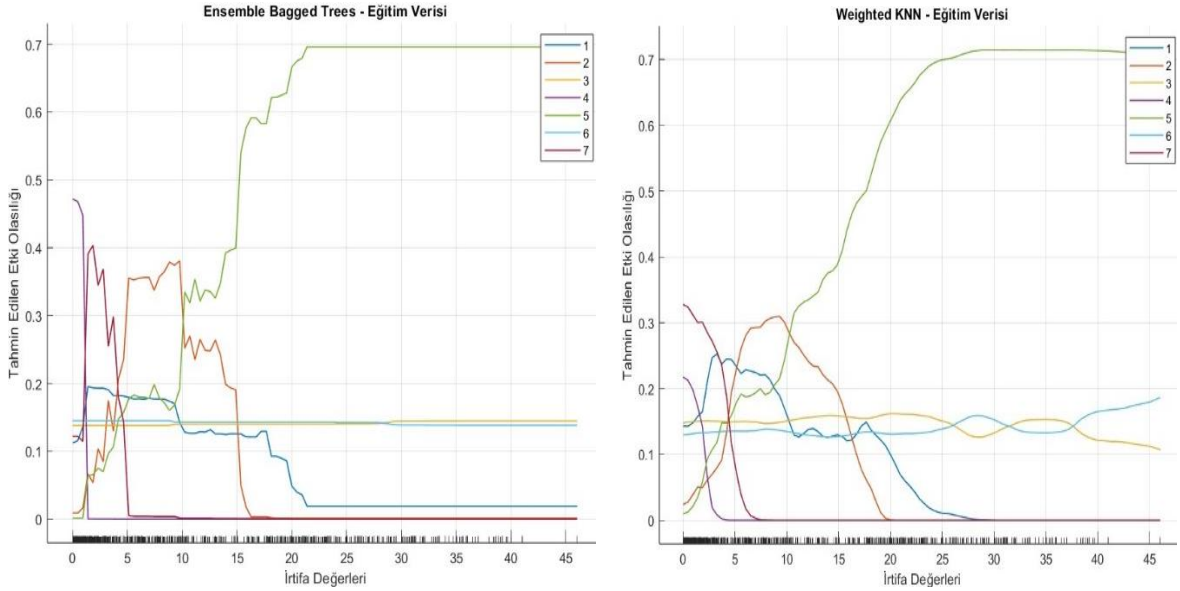
Hız verisi için Şekil 5.5'te sunulan PDP grafiklerinde modellerin sınıf bazında gösterdiği performans:

Sınıf 1: Ensemble Bagged Trees hız değerlerinde sınıf 1'de daha sabit bir tahmin eğilimi gösterirken, Weighted KNN modelinde sınıf 1 için hız değerlerine daha hassas bir yanıt gözlemleniyor.

Sınıf 2: Her iki modelde de sınıf 2 hız artışına paralel bir tahmin artışı gösteriyor, ancak Weighted KNN modeli daha keskin bir eğilim sergileyerek sınıf 2'yi hız üzerinden daha iyi ayırt ediyor.

Sınıf 3 ve 4: Weighted KNN modelinde sınıf 3 ve sınıf 4 hız değeri değişimlerine karşı belirgin bir yanıt verirken, Ensemble Bagged Trees bu sınıflarda daha yatay bir seyir gösteriyor.

Sınıf 5-7: Ensemble Bagged Trees bu sınıflarda daha düşük hassasiyete sahipken, Weighted KNN'de hız değerleri arttıkça sınıfların birbirinden ayrılma eğiliminde olduğu görülüyor. Hız verisinde genel olarak Weighted KNN modeli sınıflar arasında daha belirgin farklılıklar yaratıyor.



Şekil 5.6. İrtifa verisinin PDP grafiği

İrtifa verisi için Şekil 5.6'da sunulan her iki modele ait PDP grafiklerinde sınıfların gösterdiği yanıt:

Sınıf 1 ve 2: Ensemble Bagged Trees bu sınıflarda irtifa değerine göre daha kararlı bir seyir izlerken, Weighted KNN modelinde bu sınıfların irtifaya karşı daha belirgin bir tahmin değişimi görülüyor.

Sınıf 3: Her iki model de irtifa değerine duyarlı, ancak Weighted KNN modelinde sınıf 3'ün tahmin değeri irtifa ile daha hızlı artıyor. Bu, sınıf 3'te Weighted KNN modelinin daha iyi sonuç verebileceğini düşündürür.

Sınıf 4-5: Weighted KNN modeli bu sınıflarda irtifa verisine bağlı olarak eğim kazanırken, Ensemble Bagged Trees sabit bir çizgi gösteriyor. İrtifa, Weighted KNN modelinde sınıf 4 ve 5 ayrımı için daha etkili görünüyor.

Sınıf 6 ve 7: Ensemble Bagged Trees modelinde irtifa değişimlerinin bu sınıflar üzerinde etkisi daha belirgin. Bu durum, yüksek irtifa verilerinde Ensemble Bagged Trees'in daha güvenilir olabileceğini gösterir.

Genel değerlendirme olarak;

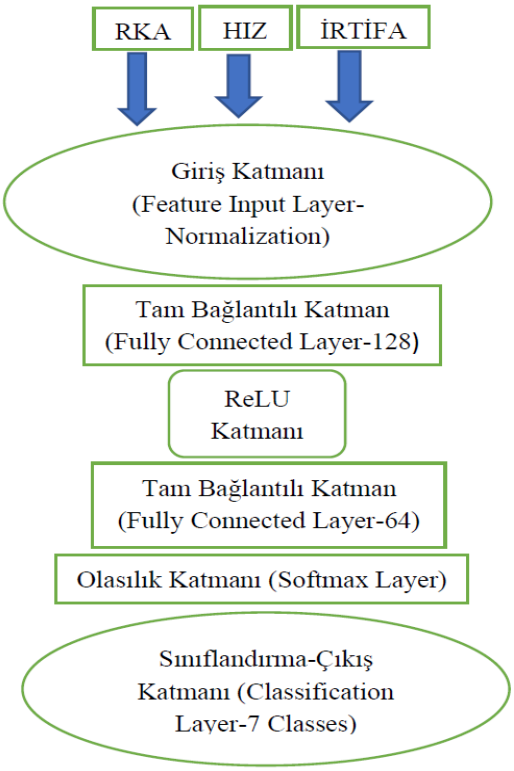
Ensemble Bagged Trees: Genellikle RKA, hız ve irtifa verilerinde daha düşük varyansla, sabit tahminler sunar. Bu model, genel sınıflandırma performansı açısından daha istikrarlı ancak bazı sınıflar için duyarlılık eksikliği gözlenebilir.

Weighted KNN Modeli: RKA, hız ve irtifa değişimlerine daha duyarlı olan Weighted KNN modeli, her sınıf için belirgin ayrımlar sergiliyor. Özellikle bazı sınıflarda Weighted KNN, daha yüksek doğruluk sağlayabilir.

5.2. Derin Öğrenmeye ait Uygulamalar

5.2.1. ANN modeli

Şekil 5.7'de ANN modelinin mimarisi içerisinde yer alan her katman, bloklar ile temsil edilerek görselleştirilmiştir. Bu katmanlar ile sunulan akış şeması dahilinde verilerin sınıflandırılmasını sağlayan bir sinir ağı modeli oluşturur. RKA, hız ve irtifa özelliklerinden her biri için derlenen 255'er adet verinin "cvpartition" fonksiyonu ile %80'i eğitim, %20'si doğrulama seti olarak ayrılması sonrasında giriş katmanına aktarılan verilerin ilk önce format uyarlanması için transpoze edilir, sonrasında normalizasyonu sağlanır ve sinir ağına giren bu verilere sırasıyla gizli katmanlar olarak tanımlanan 128 nöronlu Tam Bağlantılı Katman, ReLU Aktivasyon Katmanı ve 64 nöronlu Tam Bağlantılı Katman basamakları üzerinden işlem yapılır. Son olarak Softmax ve Classification katmanları modelin çıkışını sağlar.



ANN Akış Şeması (Metinsel Tanım)

1. **Giriş Verileri (**RKA, Hız, İrtifa)
2. **Giriş Katmanı** (Veriler normalizasyon işlemine tabi tutulur ve sinir ağına aktarılır)
3. **Tam Bağlantılı Katman** (128 Nöron ile veriyi işler ve modelin karmaşık veri ilişkilerini öğrenmesini sağlar)
4. **ReLU Aktivasyon Katmanı** (Negatif değerdeki giriş verilerini eler)
5. **Tam Bağlantılı Katman** (64 Nöron ile veriyi işleyerek sonraki katmana aktarır)
6. **Olasılık Katmanı** (Sınıf tahminlerine ilişkin olasılıkları hesaplar)
7. **Sınıflandırma Katmanı** (Modelin çıktıları ile gerçek etiketler arasındaki farkı minimize etmek için ağırlıkları günceller ve nihai sınıf çıkışını yapar)

Şekil 5.7. ANN modeline ait mimari ve akış şeması

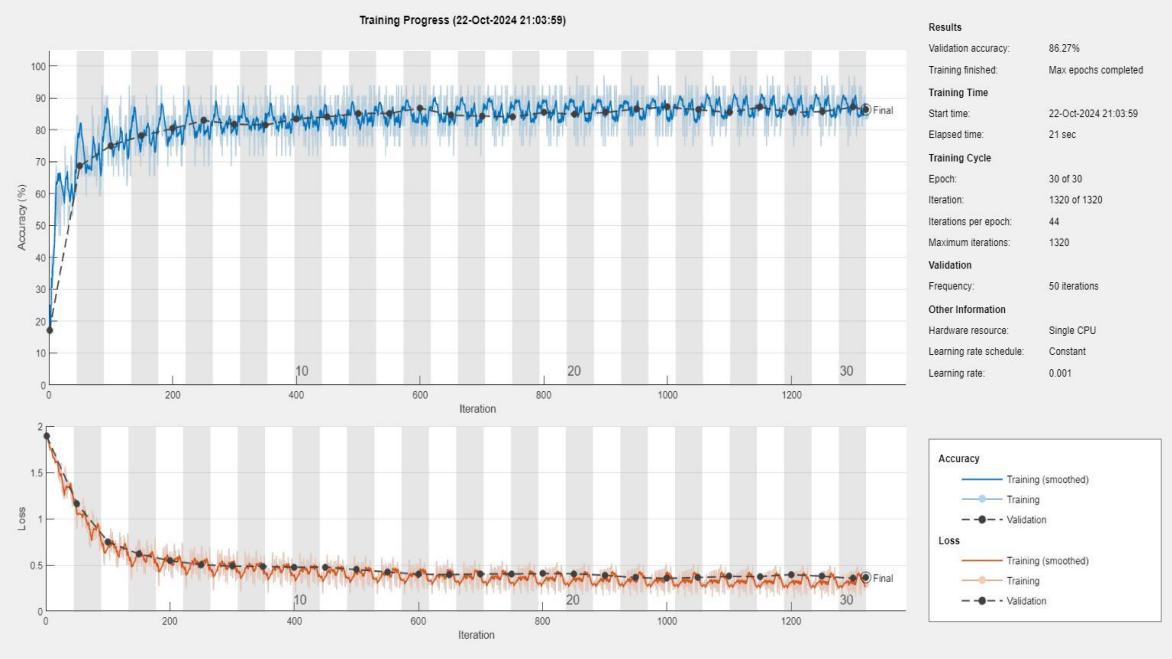
Bu modelde, 'zscore' kullanılarak verilere ait her özelliğin ortalaması 0 ve standart sapması 1 olacak şekilde normalize edilmesini müteakip verilerin belirli bir ölçek standardında olması sağlanır. Ölçeklenmiş veriler ile modelin daha hızlı ve doğru öğrenmesinin temeli sağlanmış olur.

Tam bağlantılı (fully connected) katmanında, sinir ağına gelen girişleri nöronlar aracılığıyla işleyip bir sonraki katmana aktarır. Bu katmanda her bir nöron, önceki katmandaki tüm nöronlara bağlıdır. İlk tam bağlantılı katmanda ki 128 adet nöron ile her giriş verisi için bir ağırlık ve bir bias değeri öğrenilir. Girdilerle bu ağırlık değerleri ile çarpılır, ardından bias eklenir ve sonuçlar bir sonraki katmana iletilir. Daha fazla nöron, daha fazla öğrenme kapasitesi ve daha karmaşık veri ilişkilerini öğrenme yeteneği sağlar, ancak aynı zamanda modelin aşırı öğrenme (overfitting) riskini de artırabilir.

Sonraki ReLU (Rectified Linear Unit) aktivasyon katmanında, girdiler negatifse sıfır yapılır, pozitifse girdiler olduğu gibi bırakır. Bu sayede sadece pozitif değerler ile işlem yapılır. ReLU, hesaplama açısından çok verimlidir ve bu özellikle derin ağlarda daha hızlı ve daha etkili öğrenme sağlar.

Softmax katmanında, çıktı olarak elde edilen ham değerler olasılıklara dönüştürür. Softmax fonksiyonu, her bir sınıf için bir olasılık hesaplar ve bu olasılıkların toplamı 1 (bir) olur. Bu, modelin bir girdiyi belirli bir sınıfa atama olasılığını gösterir. Özetle bu katman, modelin tahminlerini sınıflara ait olasılıklara çevirir. Model, en yüksek olasılığa sahip sınıfı tahmin eder.

Son olarak, sınıflandırma katmanı olan classification layer'de modelin tahminleri ve gerçek etiketler arasındaki kaybı hesaplar. Kaybı minimize etmek, modelin daha doğru tahminlerde bulunmasını sağlar. Bu katmanda kullanılan kayıp (loss) fonksiyonu (cross-entropy loss) tahmin edilen sınıf olasılıkları ile gerçek sınıf etiketleri arasındaki farkı ölçer. Bu kayıp, sinir ağı boyunca geri yayılır (backpropagation) ve bu farkı minimize etmek için ağırlık değerleri güncellenir. Sonucunda nihai sınıf tanımlamaları için gerekli model tasarlanmış olur.



Şekil 5.8. ANN modeli eğitim aşaması

Şekil 5.8'de yer alan grafikte, ANN modeline ait eğitim sürecindeki doğruluk (accuracy) ve kayıp (loss) değerlerinin değişimi gösterilmiştir. Modelin eğitim aşamalarında nasıl performans gösterdiğini anlamak ve modelin doğru bir şekilde öğrenip öğrenmediğini değerlendirmek amacıyla grafikler üzerinde inceleme yapılabilir.

Üstteki grafik, eğitim (training) ve doğrulama (validation) verisi üzerinde elde edilen doğruluk yüzdesini (accuracy) göstermektedir. Doğruluk, modelin doğru tahmin yapma oranıdır ve %100'e yaklaştıkça modelin performansının arttığını gösterir. Alttaki grafik,

modelin eğitim ve doğrulama verisi üzerindeki kayıp değerini (loss) gösterir. Kayıp değeri, modelin tahminlerinin ne kadar hatalı olduğunu ifade eder ve 0'a yaklaştıkça modelin doğruluğunun arttığını gösterir. Eğitim ve doğrulama kayıp grafiği zamanla azalmalıdır.

Aynı zamanda, bu grafikler, modelin aşırı öğrenme (overfitting) veya eksik öğrenme (underfitting) yapıp yapmadığını tespit etmek için incelenebilir. Örneğin, doğrulama doğruluğu düşerken eğitim doğruluğu yüksekse, model eksik öğreniyor olabilirken, eğer doğrulama kaybı eğitimin aksine bir noktada artmaya başlarsa, bu aşırı öğrenme işareti olabilir.

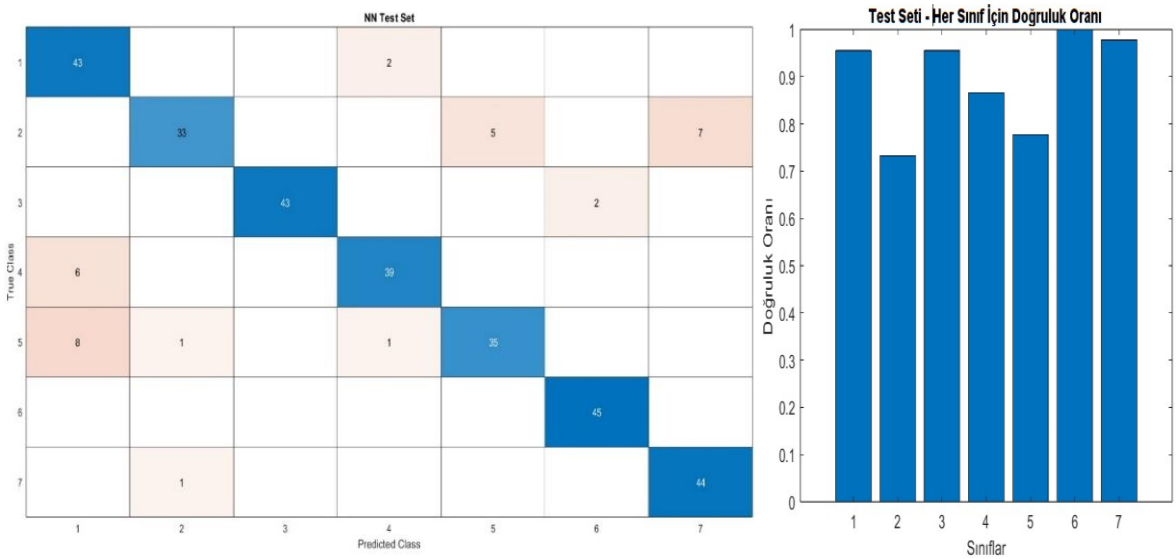
Bu grafikler, modelin eğitim sürecini optimize etmek ve gerekli ayarlamaları yapmak için önemli bilgiler sunmaktadır. Bu eğitim sürecinde doğrulama doğruluğu %86,27 olarak gözlemlenmiş, ki bu değer modelin makul bir performans gösterdiğini işaret etmektedir.

Grafikler üzerinde görebildiğimiz diğer bir adım ise Training Cycle (Eğitim Döngüsü) aşamalarıdır. Bu başlığın altında yatan epoch tanımı, eğitim verisinin tamamının modele kaç kez gösterildiği anlamına gelir. Her epoch sonunda model, tüm eğitim verisi üzerinden bir güncelleme yapar. Söz konusu başlık atında yer alan iteration (yineleme)**,** bir mini-batch'in (küçük veri parçasının) model üzerinde çalıştırılma işlemidir. Bu çalışmada eğitim 30 epoch boyunca yapılır ve mini-batch boyutu 32'dir. Bir epoch sırasında model, veri setini mini-batch'lere böler ve her birini sırayla işler. Bu grafikte, her epoch 44 iterasyondan (yineleme) oluşmaktadır ve toplamda 1320 iterasyon gerçekleştirilmiştir.

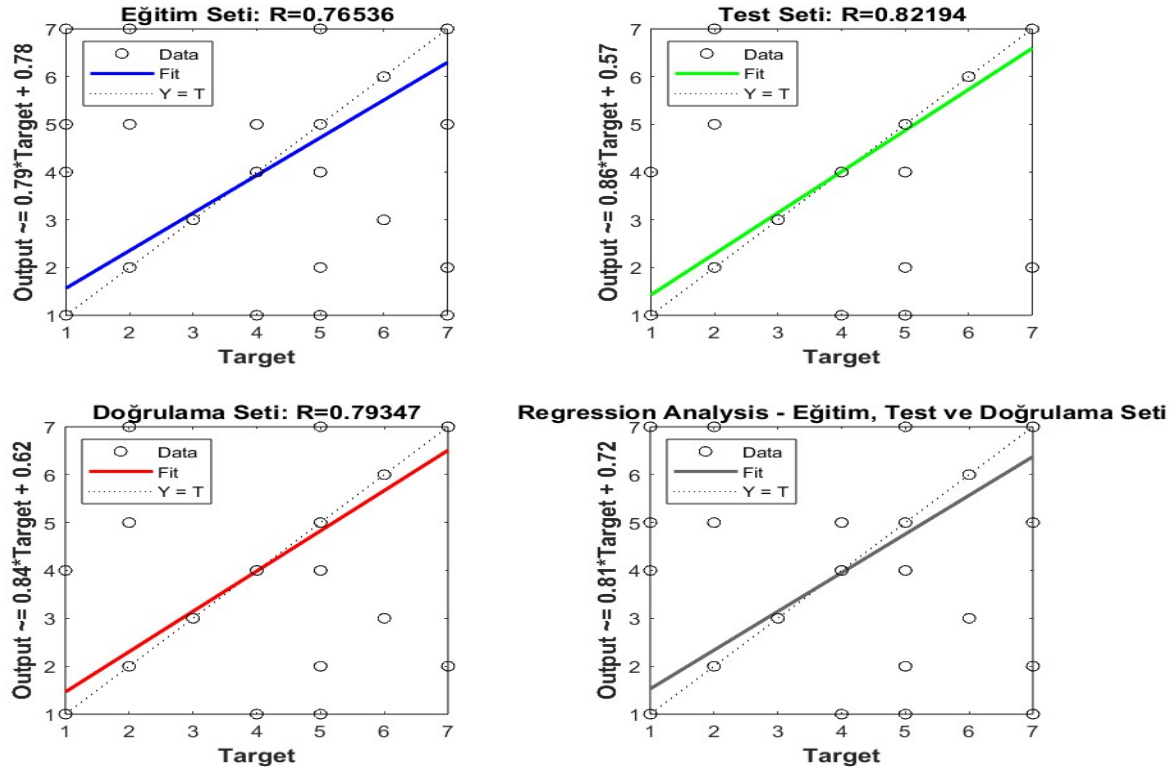
Frequency (sıklık) değeri ile doğrulama seti üzerinde modelin performansını test etme sıklığını ifade edilmektedir. Bu grafikte, her 50 iterasyonda bir doğrulama yapılmıştır. Bu sayede, modelin her 50 iterasyonda doğrulama verisi üzerindeki performansı gözlemlenmiştir.

Learning Rate (Öğrenme Hızı) değeri ile modelin her güncelleme sırasında parametrelerini ne kadar değiştireceğini belirler. Düşük bir öğrenme hızı, modelin daha yavaş öğrenmesine sebep olur ancak daha hassas sonuçlar elde edilmesini sağlar. Yüksek bir öğrenme hızı ise daha hızlı öğrenme sağlar ancak modelin doğru bir noktada durmamasına sebep olabilir. Bu grafikte, öğrenme hızı "constant" (sabit) kalacak şekilde 0.001 olarak ayarlandığı görülmektedir.

Şekil 5.9'de sunulan karışıklık matrisi ve başarı oranı grafiği ile oluşturulan modelde denenen ve her sınıf için 45 adet veriden oluşan test setinin işlem sonucunu görmekteyiz. Daha önce yapılan eğitim sürecinde işlenmeyen verilerin başarı yüzdeleri göz önünde bulundurulduğunda, en başarılı sınıfın %100 ile 6'ncı sınıf (savaş uçağı), başarı oranının en düşük olduğu sınıfın ise %73,33 ile 2'nci sınıf (helikopter) olduğu görülmektedir. Toplamda ise %89,52 başarı oranı elde edilmiştir.



Şekil 5.9. ANN modeli test verisi başarı oranı



Şekil 5.10. ANN modeli regresyon analizi

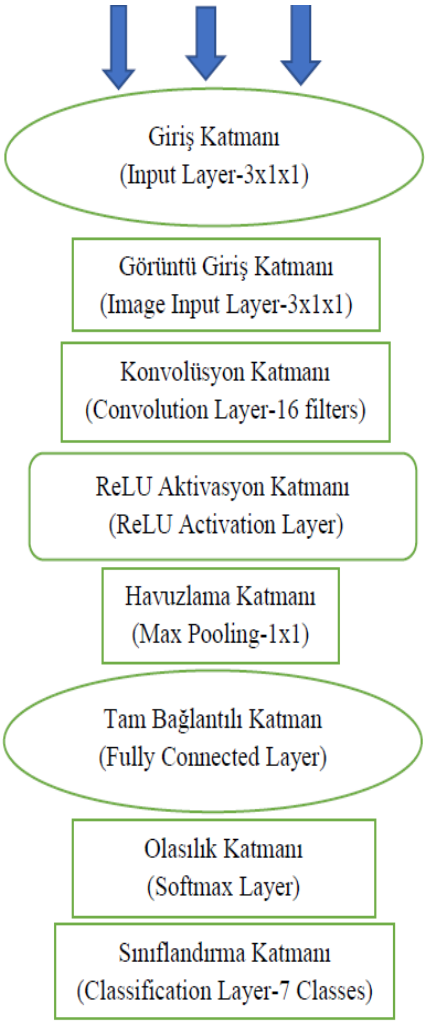
Modelin performansını görselleştirmek için yapılan diğer bir uygulamada regresyon analizidir. Şekil 5.10'da sunulan ANN modeline ait analizde gerçek ve tahmin edilen değerler (modelin çıktıları) arasındaki ilişkiyi gösterilmektedir. Output tanımı modelin tahmin ettiği, target tanımı ise gerçek hedef değerlerini karşılamaktadır. Tahmin edilen değerlerin gerçek hedeflerle birebir aynı olduğu ideal çizgi Y = T çizgisi ile belirtilmektedir. Fit Çizgisi ise modelin eğitim, test veya doğrulama setinde tahmin ettiği sonuçların doğrusal regresyonudur. Her alt grafikte, hedefe en yakın tahminin olduğu fit eğrisi ile modelin (Y=T) doğrusu karşılaştırılır. Eğitim, test, doğrulama ve tüm verileri için yapılan bu analizde kullanılan R değeri, modelin tahmin doğruluğunu ölçer. R Değeri, modelin doğruluğunun bir ölçüsü olan korelasyon katsayısıdır. Değer 0 ile 1 arasında değişir; 1'e yakın olması, tahminler ile hedefler arasında güçlü bir doğrusal ilişki olduğunu gösterir. Bazı çalışmalarda R değerinin yanı sıra determinasyon katsayısı olan R^2 değerinin de incelendiğini görmekteyiz. Her iki değerde, modelin tahmin edilen ve gerçek değerler arasındaki ilişkiyi ve performansı ölçer, fakat farklı açılardan değerlendirirler. Modelin tahmin doğruluğunu ölçmek için kullanılan iki önemli istatistiksel değerden birisi olan R^2, modelin tahmin ettiği varyans oranını göstermek amacıyla R'nin karesi olarak hesaplanır. Genellikle regresyon analizinde modelin veri setindeki varyansı ne kadar açıkladığını anlamak için kullanılır. Regresyon analizinde varyans, modelin tahmin edilen değerler ve gerçek değerler arasındaki farkların karelerinin ortalaması olarak ölçülür. Model varyansı bir modelin tahminlerinde ne kadar dalgalanma olduğunu ifade eder. Yani modelin çıktılarının, hedef veya gerçek değerler etrafında ne kadar "dağıldığını" gösterir. Genel olarak regresyon analizinde modelin doğruluğunu ve uyumunu anlamak için doğrudan R değeri kullanılır iken modelin tüm veri üzerindeki genel performansını değerlendirmek için ise R^2 tercih edilir. Yapılan çalışmadaki analizde elde edilen R değerlerine bakıldığında; modelin eğitim setinde 0.76'lık bir korelasyonla orta düzey başarı oranıyla tahmin yaptığı, test setinde 0.82'lik R değeri ile eğitim setine göre daha yüksek bir korelasyon sağladığı ve modelin, eğitim sonrası test verisinde eğitim verisinden biraz daha iyi performans gösterdiği görülmektedir. Test setindeki yüksek korelasyon, modelin genelleme gücünün iyi olduğunu işaret etmektedir. Doğrulama setinde R değeri 0.79 olup, eğitim setine yakın bir sonuç göstermektedir. Bu, doğrulama verisinde de hedeflere yeterli düzeyde yakın tahminler yapıldığını ancak ideal bir performansın sağlanmadığını gösterir.

Genelleme yapacak olursak, modelin test verisinde yüksek bir korelasyon elde etmesini iyi bir genel performans işareti olarak değerlendirebiliriz. Eğitim, test ve doğrulama setleri

arasındaki performans farklılıklarının çok yüksek olmamasını modelin dengeli bir şekilde öğrenmiş olduğunu gösterirken, R değerlerinin ideal seviyede olmaması, modelin iyileştirilebilir olduğunu göstermektedir.

5.2.2. CNN modeli

Çalışmamızın bu bölümünde CNN modeli kullanılarak uçuş verilerinin işlenmesindeki aşamaları ve sonucunda elde edilen performans kriterlerinin incelenmesi yapılacaktır.



CNN Akış Şeması (Metinsel Tanım)

1. **Giriş Verileri (**RKA, Hız, İrtifa)
2. **Giriş Katmanı** (Veriler normalizasyon işlemine tabi tutulur)
3. **Görüntü Giriş Katmanı (**Giriş boyutu 3x1x1 olacak şekilde ayarlanarak verinin 1x1 boyutunda ve 3 kanala sahip "bir resim" gibi işlenmesine olanak sağlar)
4. **Konvolüsyon Katmanı (**Kullandığı 16 filtre (kernel) sayesinde 3x1x1 giriş boyutu 3x1x16 boyutuna dönüştürerek 16 farklı özellik haritası (feature map) elde edilir ve CNN'nin temelini oluşturur)
5. **ReLU Aktivasyon Katmanı** (Negatif değerler sıfırlanarak modelin sadece pozitif özniteliklere odaklanması sağlanır)
6. **Havuzlama Katmanı** (Verinin özetlenmesi sağlanır)
7. **Tam Bağlantılı Katman** (7 sınıfa bağlanan tüm nöronlar ile sonraki katmanına aktarım yapar)
8. **Olasılık Katmanı** (Sınıflar arasında olasılık dağılımını sağlar)

Şekil 5.11. CNN modeline ait mimari ve akış şeması

Şekil 5.11 ile ana mimarisi gösterilen CNN modeline ait işlem adımlarının başlangıç noktasını oluşturan giriş katmanında, 3 farklı özelliğe (RKA, hız, irtifa) sahip veri seti için boyutlar 3x1x1 olacak şekilde ayarlanıyor. 3 sayısal veri, giriş katmanına alınarak 1 boyutlu bir vektör olarak uyarlanıyor. Görüntü giriş katmanında verinin CNN modeline uygun hale getirilmesi sağlanıyor. CNN yapıları genelde görüntü verisiyle çalıştığından, Reshape

fonksiyonu neticesinde 3x1x1 formatındaki bu 3 özellik modelin "görsel" olarak işlediği veriler haline dönüştürülüyor. CNN modeli, genellikle görüntüleri işlemek için tasarlanmıştır ve giriş olarak genelde 3B (yükseklik x genişlik x kanal sayısı) matrislerini temel alır. Uygulamada, 3 adet sayısal özellikten (RKA, hız, irtifa) oluşan veriyi CNN'in işleyebilmesi için belirli bir formatta yeniden şekillendirme ihtiyacı bulunmaktadır. Bu durumda, 3 özellik bir görüntünün renk kanalları gibi düşünülebilir. Yükseklik ve genişlik 1 ve kanal sayısı 3'tür (RKA, hız ve irtifa verileri 3 farklı kanal gibi kabul edilir). Böylelikle özelliklerin her biri bir kanal olarak işlenir. Bu kanallar, CNN'deki konvolüsyon ve havuzlama katmanlarında tıpkı bir görüntü gibi ele alınır. CNN'deki konvolüsyon katmanları, bu 3 veriyi birbirine bağlı olarak işleyerek her birinin farklı özelliklerini ve ilişkilerini öğrenmesi esnasında filtre adı verilen küçük matrisler (kerneller) kullanır ve giriş verisine belirli bir alan üzerinde uygulanarak bu bölgedeki bilgiyi özetler. Özetleme işlemi sayesinde, giriş verisinin önemli özelliklerini çıkartılarak modelin karmaşıklığı azalır. Bu filtrelerin her biri, giriş verisine ait farklı özellikler üzerinde kayan bir pencere gibi hareket eder ve görüntüde belirli bir desen (pattern) arar (örneğin, kenarlar, dokular).

Bu işlemler içerisinde iki farklı adım uygulanır.

Kaydırma (stride)**:** Filtrenin veri üzerinde ne kadar adım atacağıdır. Stride değerinin 1 olması, filtrenin her bir adımda bir birim kayacağını gösterir. Bu, daha ayrıntılı bilgi çıkarılmasını sağlar. Daha büyük bir stride değeri, daha hızlı ancak daha düşük çözünürlükte sonuçlar verir.

Doldurma (padding)**:** Kenardaki verileri korumak için giriş verisine sıfır ekleme işlemidir. Özellikle, verinin kenarlarında bilgi kaybını önlemek için kullanılır. "Same" padding, filtrenin boyutunu koruyarak giriş verisinin aynı boyutla devam etmesini sağlar. Giriş ve çıkış boyutlarını aynı tutmak için kenar dolgusu eklenir.

Yapılan işlemler neticesinde oluşan çıktıda, farklı özellik haritaları (feature maps) üretir ve sonraki katmanlara iletir. Bu uygulamada 16 filtre kullanıldığında çıkış boyutu 3 x 1 x 16 olur.

Konvolüsyon işlemi tamamlandıktan sonra ReLU aktivasyon fonksiyonu uygulanır. ReLU, negatif değerleri sıfıra çekerken yani giriş verisinin negatif kısmını kırparken, pozitif değerleri olduğu gibi bırakır. Öğrenme sürecini hızlandırmak ve doğrusal olmayan

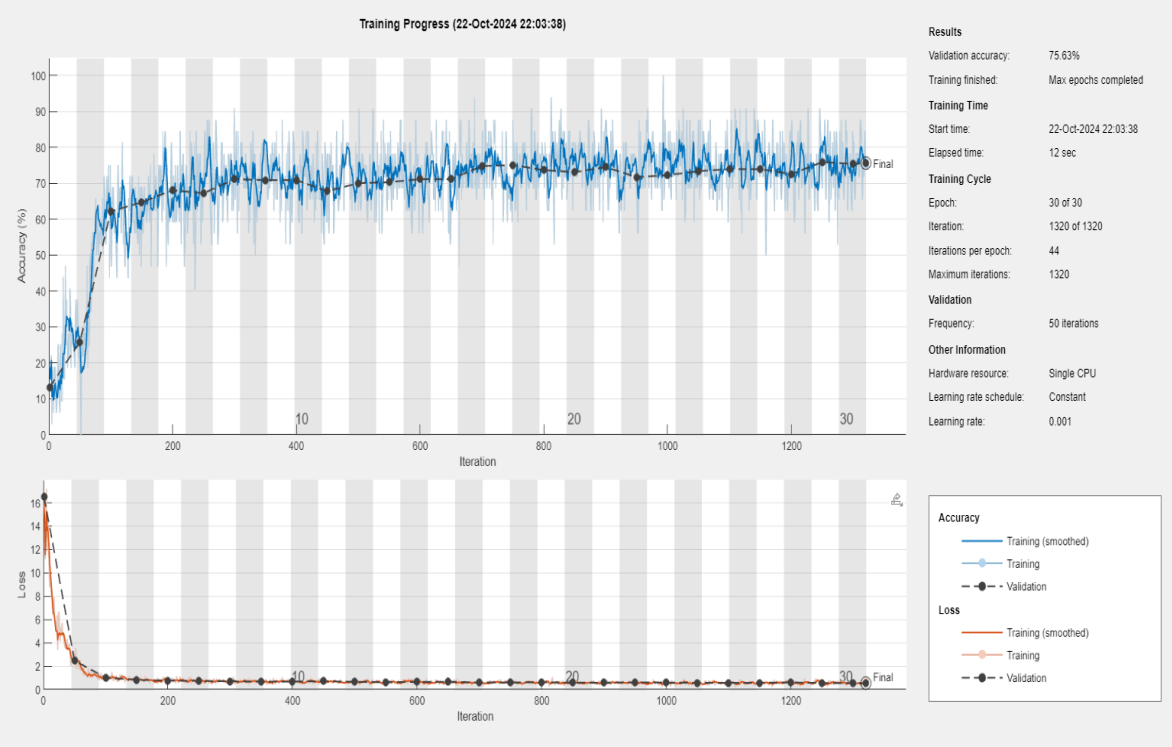
dönüşümlere olanak sağlamak amacıyla kullanılan ReLU fonksiyonu ile modelin daha karmaşık ilişkileri öğrenmesine olanak sağlanır.

Bu işlemden sonra, havuzlama **(**max-pooling**)** katmanı kullanılır. Bu katmanın amacı, veri boyutunu küçülterek önemli özelliklerin korunmasını sağlamaktır. Max-pooling fonksiyonu, belirli bir bölgedeki en büyük değeri alır ve bu şekilde veri özetlenir. İşlem yaparken her küçük bölgedeki en büyük değeri seçer. Böylelikle veri boyutu azalırken (çözünürlük düşerken) hem işlem hızı artar hem de modelin genelleme kapasitesi yükseltir. Havuzlama, gereksiz ayrıntıları eleyerek modelin daha genel öznitelikler öğrenmesine yardımcı olur. Böylelikle model daha hızlı çalışır ve verinin ayrıntılı tutulması durumunda karşılaşılan aşırı öğrenme (overfitting) sorununun önüne geçilir. Ancak bu çalışmada havuzlama işlemi boyutu değiştirmediği için burada 1x1'lik küçük bir bölge üzerinden işlem yapılıyor. Bu ağda, havuzlama işlemi boyut küçültme yerine özellik haritalarındaki en önemli bilgiyi korumak amacıyla kullanılmıştır. 1x1 havuzlama, herhangi bir küçültme yapmadığı için seçilmiştir.

Tam bağlantılı katmanda giriş veriler düz bir vektöre dönüştürür ve her örneğin 7 sınıftan birine atanması için gerekli bağlantılar kurulur. Bu katman, bir nöronlar dizisidir ve her nöron bir sınıfla ilişkili olacak şekilde daha önceki katmandan gelen bilgileri kullanır. Her bir sınıfın ağırlıklandırılmış çıkışları hesaplanarak olasılık katmanına aktarılır.

Softmax fonksiyonu her sınıfın olasılık dağılımını sağlar ve toplamı 1 (bir) olan bir olasılık vektörüne dönüştürür. Modelin hangi sınıfa daha yakın olduğunu gösterir.

Son katmanda ise sınıflandırma (classification) fonksiyonu, nihai olarak sınıf tahmini işlemini yapar. En yüksek olasılığa sahip olan sınıf ataması çıkış olarak döndürülür. Bu nedenle, Tam bağlantılı katmandaki nöron sayısı, sınıflandırma yapılacak sınıf sayısına eşittir. Modelin tahmin ettiği sınıflar ile gerçek sınıflar karşılaştırılır ve modelin başarısı ölçülür.

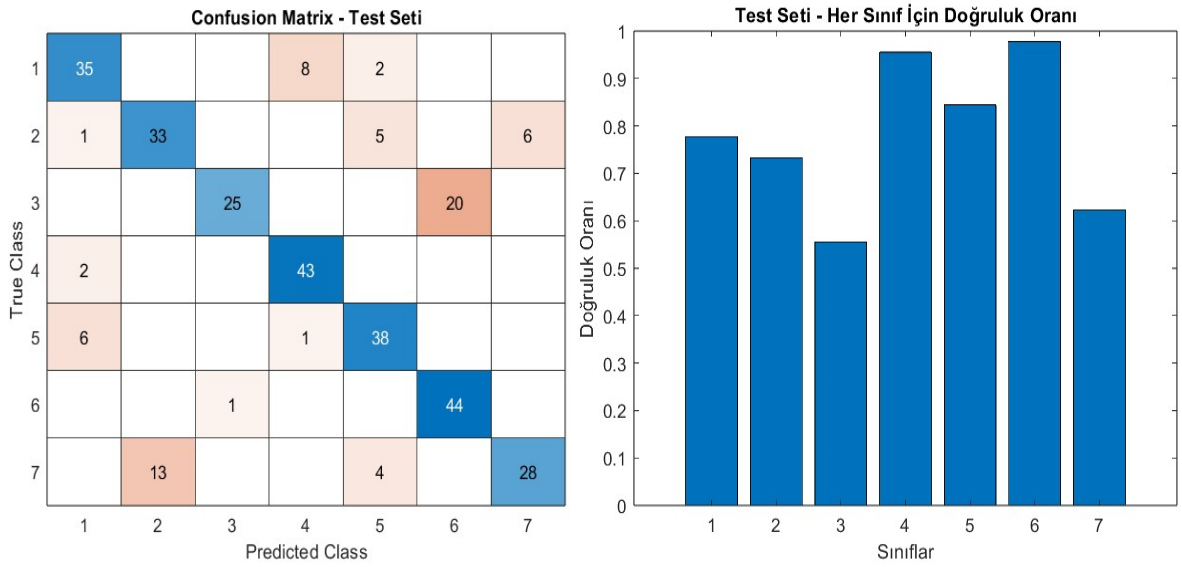


Şekil 5.12. CNN modeli eğitim aşaması

Şekil 5.12'de yer alan CNN modelinin eğitim aşamasına ait grafiklerde, eğitim ve doğrulama doğruluk çizgisinin genel olarak yükseldiğinden modelin öğrenme hedefini gerçekleştirdiğine ve doğrulama doğruluğu ile eğitim doğruluğunun eşgüdüm hareketinden ise modelde aşırı öğrenme veya eksik öğrenme sorunlarının olmadığına dair değerlendirmeler yapılabilir.

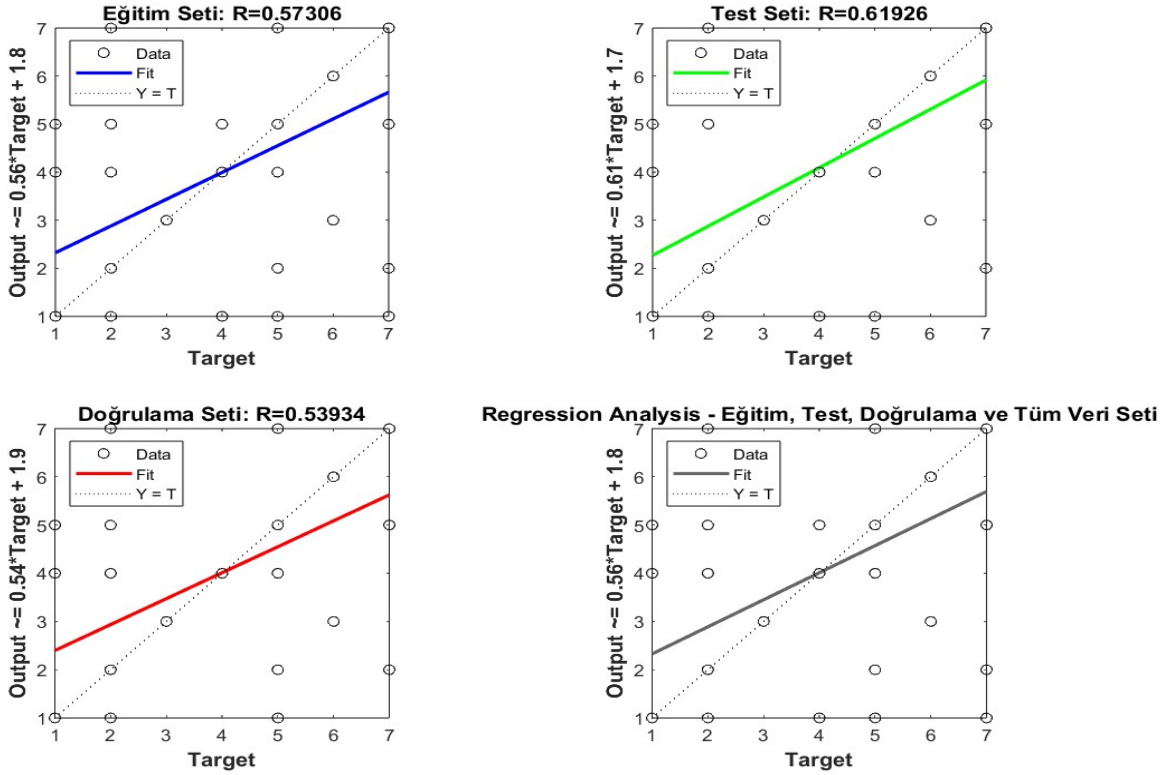
Hem eğitim hem doğrulama verileri için doğruluk grafiği zamanla artış göstermiş, sürecinin ilk birkaç epoch'unda doğruluğun hızlı bir şekilde arttığı ve ardından daha stabil hale geldiği görülmektedir.

Eğitim ve doğrulama doğrulukları arasında önemli bir farkın olmaması modelin eğitim verisine aşırı odaklanmadığını ve doğrulama verisinde de başarılı olduğunu göstermektedir. Diğer taraftan, eğitim ve doğrulama kayıp grafikleri de zamanla azalmış ve her iki grafik birbirine yakın seyretmiştir. Benzer şekilde, bu tespit ışığında modelin doğrulama setinde de eğitim setinde olduğu kadar iyi performans gösterdiği ve aşırı öğrenme belirtisi olmadığı görülmüştür.



Şekil 5.13. CNN modeli test verisi başarı oranı

Şekil 5.13'te sunulan karışıklık matrisi ve başarı oranı grafiği ile oluşturulan modelde denenen ve her sınıf için 45 adet veriden oluşan test setinin sonucunu görmekteyiz. Daha önce yapılan eğitim sürecinde işlenmeyen, modelin ilk kez karşılaştığı verilere ait başarı yüzdeleri göz önünde bulundurulduğunda, en başarılı sınıfın %97,77 ile 6'ncı sınıf (savaş uçağı), başarı oranının en düşük olduğu sınıfın ise %55,55 ile 3'üncü sınıf (yolcu uçağı) olduğu görülmektedir. Toplamda ise %78,09 başarı oranı elde edilmiştir.



Şekil 5.14. CNN modeli regresyon analizi

Şekil 5.14'te sunulan regresyon analizine ait alt grafiklerde, modele ait eğitim, test, doğrulama ve tüm veri setleri için performans değerlendirilmesi yapılabilir. Şöyle ki;

a. Eğitim Seti (R=0.57306)

R değerine yönelik, modelin hedefler ile tahminler arasında orta derecede bir ilişki olduğu, eğilim çizgisine yönelik ise, modelin tahminlerinin hedefe yakınsamakla birlikte doğrusal bir ilişki kurmakta zorluk çektiği görülmektedir.

b. Test Seti (R=0.61926)

Test setindeki R değerinin, eğitim setine kıyasla biraz daha yüksek bir korelasyon gösterdiği, bu doğrultuda, modelin test verileri üzerinde eğitim verilerine kıyasla daha iyi bir performans sergilediği, regresyon çizgisi eğiminin (0.61) ise, eğitim setine yakın ancak hedefe ulaşmada hala eksik kaldığı anlaşılmaktadır.

c. Doğrulama Seti (R=0.53934)

Doğrulama setindeki R değerinin, test ve eğitim setlerine nazaran biraz daha düşük olmasına bağlı olarak modelin doğrulama verileri üzerinde diğer iki set kadar iyi performans göstermediği; regresyon doğrusuna yönelik ise eğimin R değerinde olduğu gibi diğer veri

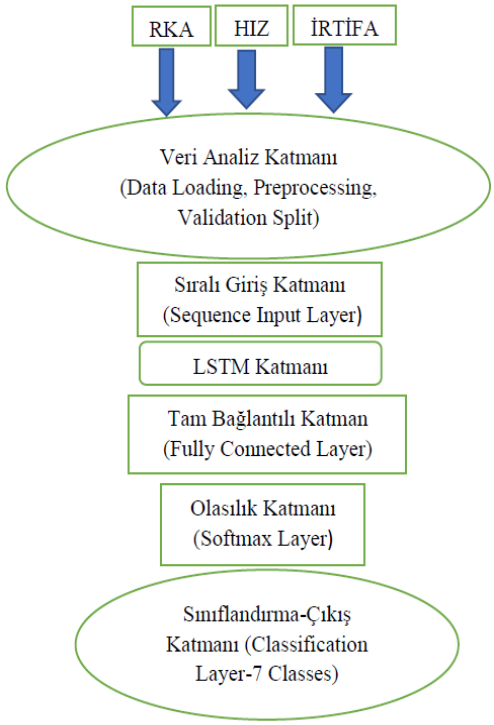
setlerinden daha düşük elde edilmesini, modelin doğrulama verilerindeki tahminlerinde hata payının biraz daha yüksek olacağına dair yorum yapılabilir.

d. Tüm Veri Seti (R=0.56006)

Tüm veri seti için R değerine bakıldığında, genel olarak modelin tahmin performansının ortalamanın biraz altında olduğu, regresyon doğrusu özelinde yapılan incelemede ise, modelin tahminlerinin hedef etiketleri ile yüksek oranda örtüşmediği anlaşılmaktadır.

Genel olarak regresyon analizi incelendiğinde, her üç set için R değerleri yaklaşık 0.5-0.6 civarında olduğu, bunun performans değerlendirilmesindeki karşılığının modelin hedef sınıflarını kısmen tahmin edebildiği, yani, oldukça yüksek bir doğrulukla tahmin yapamadığı; veri setlerindeki regresyon çizgilerinin ve eğim değerlerinin birbirine yakın, ancak hedeflere ait doğrusal ilişkinin tam olarak sağlanamadığı görülmektedir. Özetle; bu analiz, modelin orta derecede bir başarı sağladığını ve daha fazla iyileştirmeye ihtiyaç duyduğunu göstermektedir.

5.2.3. RNN modeli



RNN Akış Şeması (Metinsel Tanım)

1. **Giriş Verileri (**RKA, Hız, İrtifa)
2. **Veri Analiz Katmanı** (verileri sıralı hücre dizilere dönüştürülerek kaydeder)
3. **Sıralı Giriş Katmanı** (Sıralı-sequence verilerin işlenebilir olması için gerekli dönüşümleri yapar)
4. **LSTM Katmanı** (100 nöron ile zaman serisi verisini işler ve sıralı bağımlılıkları öğrenerek anlamlandırır)
5. **Tam bağlantılı Katman** (Gelen veriyi sınıf tahmini yapabilmek amacıyla işleyerek nöron sayısına göre boyutlandırır.
6. **Olasılık Katmanı** (Sınıflar arasında olasılık dağılımı hesaplar.)
7. **Sınıflandırma Katmanı** (Modelin çıktısını her sınıf için bir olasılık dağılımına göre nihai sınıflandırmayı yapar)

Şekil 5.15. RNN modeline ait mimari ve akış şeması

Derin Öğrenme yöntemlerinden incelediğimiz diğer bir model ise RNN (Recurrent Neural Networks-Tekrarlayan Sinir Ağı)'dir. Şekil 5.15'de verilen mimari ve akış şeması ile takip edilen fonksiyonel adımlar özetlenmiştir.

Verilerin modele ilk girdiği yer olan Veri Analiz Katmanında, veri yükleme, ön inceleme ve doğrulama verilerinin ayarlanma işlemleri yapılmaktadır. Sonraki aşamada analiz edilen veriler Sıralı Giriş Katmanına aktarılmaktadır. RNN yapılarında kullanılan bu girdi katmanında, modelin zaman serisi verisini alması ve işlenmesi sağlanmaktadır. Bu katman, modelin veri kümesindeki zaman adımları ve özellik sayısını alarak her adımda bir dizi veri örneği (özellik vektörü) işler. Veri kümesi (özellik sayısı x zaman adımı sayısı) boyutundayken, Sequence Input Layer her zaman adımında bu özellik vektörünü LSTM (Long Short-Term Memory) katmanına geçirir. Böylelikle LSTM katmanına sıralı bir şekilde özellik vektörleri sağlanır, bu vektörler her bir zaman adımında LSTM'nin girişine gider. Çıktının boyutları (inputSize, batchSize, timeSteps) formatındadır. Her zaman adımındaki inputSize özellik vektörünü, batchSize örnek gruplarını, timeSteps ise zaman adımlarını temsil eder.

inputSize (Girdi Boyutu)**:** Bu boyut, her bir örneğin RKA, hız, irtifa gibi özelliklerden oluşan özellik vektörünün uzunluğunu tanımlar. Örneğin, her bir girdinin 3 özelliği varsa, inputSize değeri 3 olur. Bu, modelin her zaman adımında aldığı bilgi miktarını ifade eder.

batchSize (Yığın Boyutu): Bu değer, modelin bir eğitim adımında işlediği örnek sayısını temsil eder. Oluşturulan kodda, MiniBatchSize değeri 32 olarak belirlendiği için, bir yığın boyutunda 32 örnek bulunur. Bu, paralel olarak işlenen veri miktarını artırır ve eğitim sürecini hızlandırır.

timeSteps (Zaman Adımları): RNN'lerde zaman adımları, veri serisinin zamansal bileşenlerini yakalamak için kullanılır. Her bir zaman adımında RNN, bir örneğin yeni bir veri noktasını işler, bu nedenle zaman adımı, dizinin her bir anlık özellik güncellemesini temsil eder. Koda bağlı olarak her bir örnek, belirli sayıda zaman adımında modellenir; yani, her zaman adımında inputSize kadar bilgi işler.

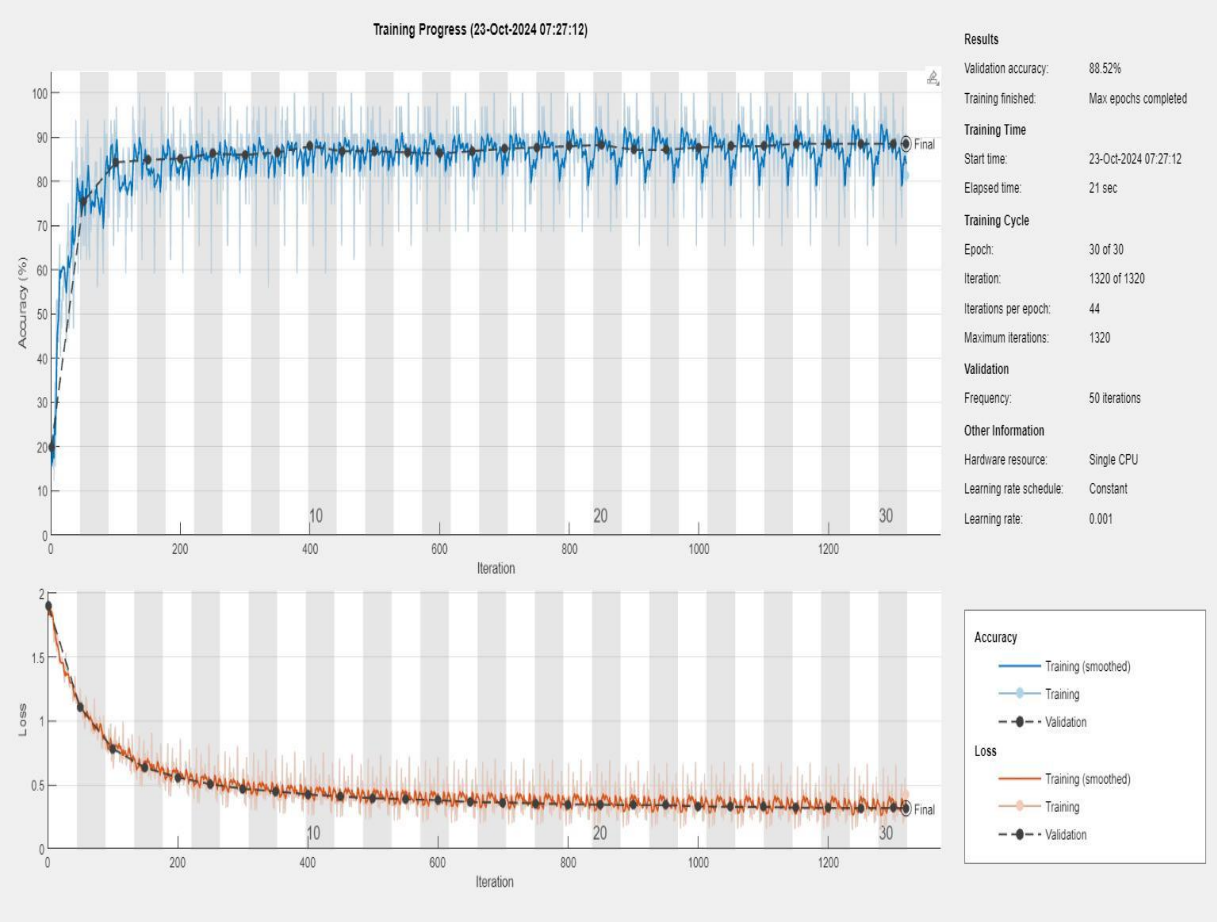
Yukarıda bahsedilen akış doğrultusunda, bu katman, ardışık veri dizilerini modelin anlayabileceği bir formata dönüştürerek LSTM katmanına iletir.

LSTM Katmanı dahilinde 100 nöron kullanılmıştır. Bu katmanda, zaman serisi verilerindeki uzun vadeli bağımlılıkları öğrenir. Bu bağımlılıklar, dizilerdeki ardışık bilgilerin önemini korumak için kritiktir. LSTM hücreleri, gate mekanizmaları (giriş, unutma ve çıkış kapıları) kullanarak sıralı veri içindeki önemli bilgileri korur. Bu kapılar verinin hangi kısmının hafızada tutulacağını veya unutulacağını kontrol eder. Bu çalışmada tercih edilen OutputMode='last' yapısı, yalnızca son çıktıyı döndürerek her bir örneğin tüm zaman adımlarını işlemesine rağmen yalnızca son durumu dikkate alır. Bu tercih, zamanla sıralı bir diziye bakıp yalnızca son durumdaki sonucu anlamak istediğimizde, genellikle sınıflandırma veya dizi tamamlanması gibi tek çıktıya ihtiyaç duyulan görevlerde idealdir. Yani, tüm zaman adımlarında LSTM katmanı sırayla veriyi işler ve son zaman adımına ulaştığında bu son çıktıyı sonraki katmana iletir.

Tam Bağlantılı Katman (Fully Connected Layer) içerisinde sınıf sayısı toplam 7 olduğu için 7 adet nöron kullanılmıştır. Her sınıf için bir nöron olması, modelin her bir sınıfa bir olasılık tahmini yapabilmesini sağlar. Sınıf sayısına bağlı olarak nöron sayısının ayarlanması, çıktı boyutunu doğrudan sınıflarla eşleştirir. Bu katman, LSTM'den gelen özellikleri kullanarak sınıflandırma tahminlerini yapar. LSTM'den çıkan veriyi alır ve her sınıfa karşılık gelen çıktıyı üretir. Tam bağlantılı katmanlar, her nöronun diğer tüm nöronlarla bağlantılı olduğu bir yapıdadır. Bu, modelin öğrendiği bilgileri sınıflandırma amaçlı işlemesine olanak tanır.

Olasılık Katmanı içerisinde softmax fonksiyonu her sınıfa ait olasılık tahmini sağlar. Sonuçlar, toplamları 1 (bir) olacak şekilde normalize edilir. Sınıflandırma problemleri için yaygın olarak kullanılan softmax fonksiyonu her bir sınıfa ilişkin çıktı değerlerini olasılık dağılımına çevirir. Yani, olasılık bazlı sınıf tahmini sağlar ve eğitim sırasında modelin doğru sınıfa yakınsamasına yardımcı olur.

Sınıflandırma Katmanı (Classification Layer) dahilinde ise modelin tahmin edilen sınıf etiketleriyle gerçek etiketleri karşılaştırması sağlanır ve bir hata hesaplaması yapılır. Kategori tabanlı bir sınıflandırma katmanı, sınıfların doğruluk oranlarını değerlendirmek için kullanılır. Bu katman, modelin hata fonksiyonunu optimize etmek için kritik öneme sahiptir.



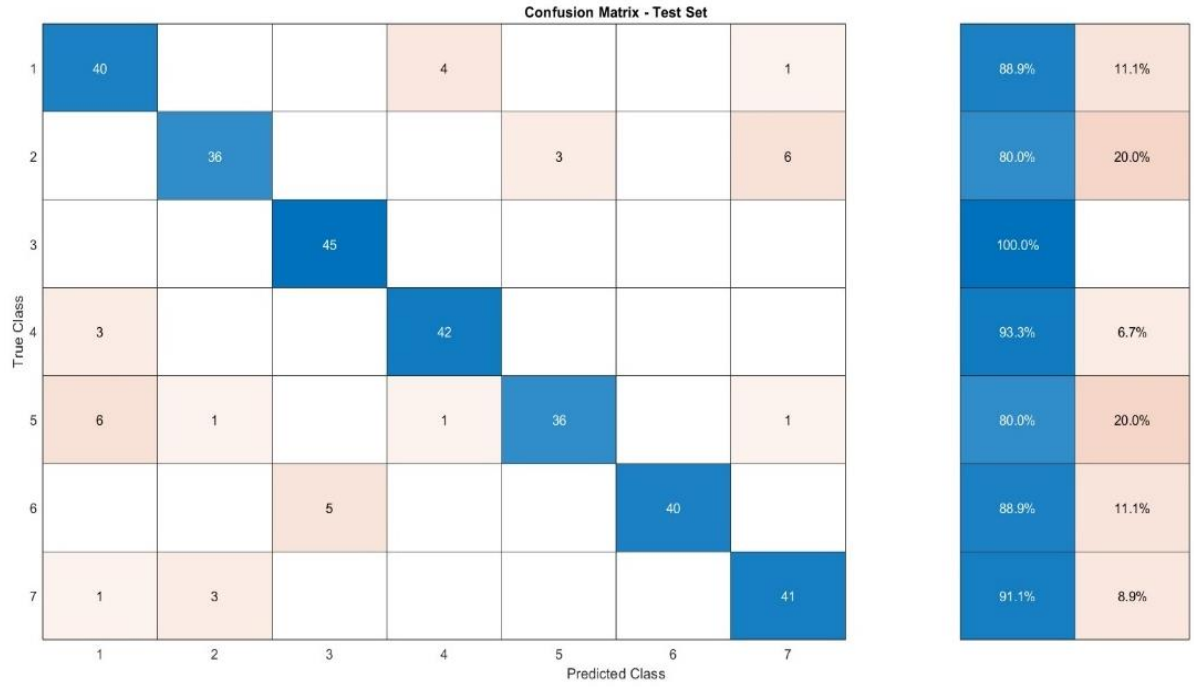
Şekil 5.16. RNN modeli eğitim aşaması

Şekil 5.16'da RNN modeline ait Eğitim aşaması ve eğitim parametreleri gösterilmiştir. Adam optimizasyon algoritması ile 30 epoch (modelin veri setini kaç kez işleyeceği bilgisi) boyunca eğitim yapılmıştır. Her adımda model 32 veri **(**Mini Batch Boyutu) örneğini işler.

Eğer model aşırı öğrenme yapıyor olsaydı, doğrulama doğruluğu bir noktadan sonra düşmeye başlayabilir ve doğrulama kaybı yükselirken, eğitim doğruluğu %100'e yaklaşabilirdi. Ancak burada eğitim ve doğrulama doğrulukları birbirine oldukça yakın seyrediyor ve benzer şekilde kayıp değerleri de birbirine paralel bir şekilde azalıyor. Eksik öğrenme durumunda ise hem eğitim hem de doğrulama doğruluğu düşük kalır ve kayıp değerleri yüksek olurdu. Bu grafikte doğrulama doğruluğu %88,52'ye ulaşmış durumda, bu da modelin veri setini makul bir şekilde öğrendiğini gösteriyor.

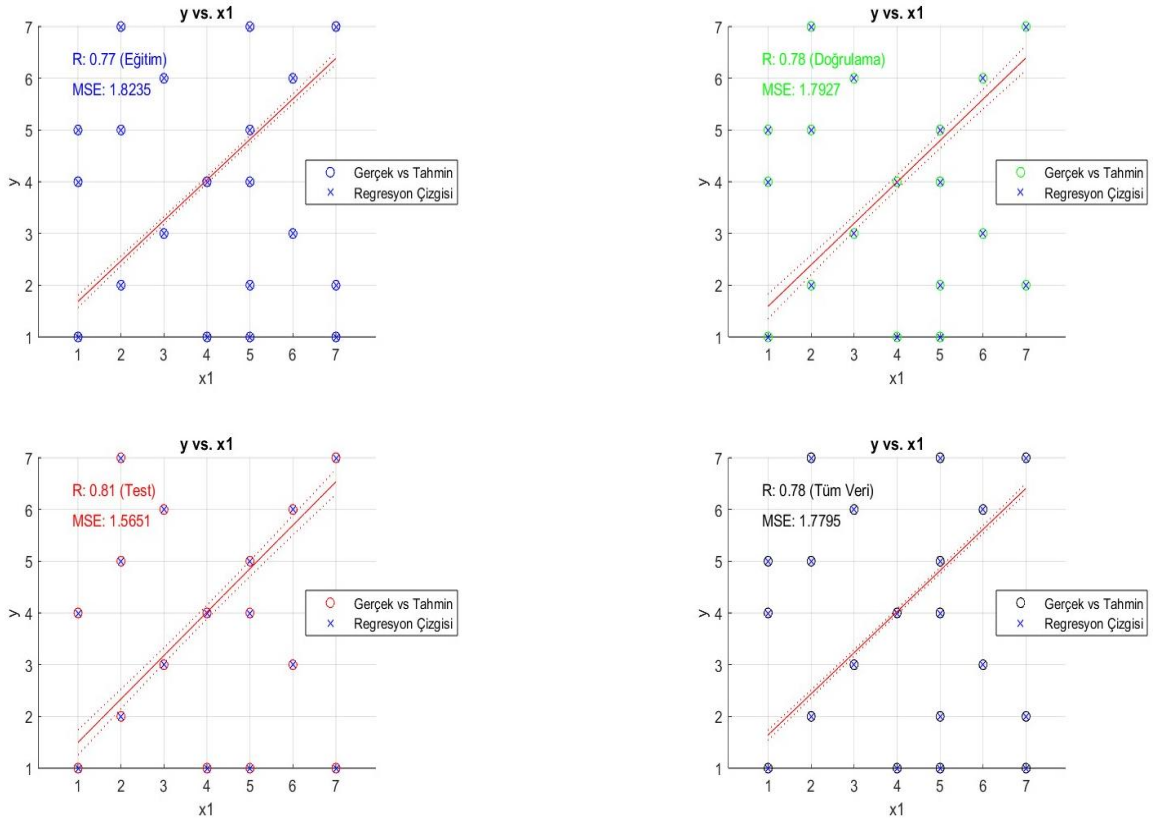
Iterasyon sayısı, bir epoch içinde kaç mini-batch kullanılacağına göre belirlenir. Mini-batch boyutu seçildiğinde, epoch başına iterasyon sayısı otomatik olarak belirlenmiş, bu sayılarda önceki derin öğrenme yöntemleri (ANN ve CNN) ile aynı değerler tercih edilmiştir.

Sonuç olarak, bu grafiğe göre modelde aşırı öğrenme veya eksik öğrenme gözlemlenmiyor. Eğitim ve doğrulama doğrulukları ve kayıpları birbirine yakın değerlerde seyrediyor, bu da modelin dengeli bir şekilde öğrenmiş olduğunu gösteriyor. Model, eğitim verisine aşırı odaklanmadan doğrulama verisinde de başarılı sonuçlar elde ediyor.



Şekil 5.17. RNN modeli test verisi başarı oranı

Şekil 5.17'de test setleri için hesaplanan karışıklık matrisi ile sınıf bazında doğruluk oranları gösterilmiştir. En başarılı sınıf 3 (yolcu uçağı) ile 100 % başarı yakalanmışken sınıf 2 (helikopter) ve sınıf 5 (İHA) ile 80 % olarak başarı yakalanmıştır.



Şekil 5.18. RNN modeli regresyon analizi

Şekil 5.18'de RNN modeline ait Regresyon Analizi ve Hata Hesaplamaları sunulmuştur. Daha önceki modellerde olduğu üzere eğitim, doğrulama, test ve tüm veri setleri için scatter plotlar ile regresyon analizleri yapılmıştır. Her bir veri seti için doğrusal regresyon çizgisi çizilmiş ve MSE (Mean Squared Error-Ortalama Kare Hatası) hesaplaması ile R (Korelasyon katsayıları) alt grafikler üzerinde gösterilmiştir.

Önceki modellerde MSE hesaplaması üzerinde durulmadığından bu bölümde bu konuda ilave bilgi verilmiştir.

$$MSE=\frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{y}_i)^2 \quad (5.1)$$

Burada:

- N: Toplam veri sayısı (örnek sayısı).
- y_i : Gerçek değer (etiket) için i. örneği.
- $\hat{y}_i$: Modelin i. örnek için tahmini değeri.

Regresyon problemlerinde yaygın bir metrik olan MSE 5.1'deki formül ile hesaplanır. MSE, tahmin edilen değerler ile gerçek değerler arasındaki farkların karelerinin ortalamasını ifade eder. Modelin hatasını veya sapmasını gösterir. Başka bir ifade ile, modelin tahminlerinin hedef değerlerden ne kadar uzaklaştığını gösterir. Düşük bir MSE değeri, modelin daha doğru tahminler yaptığı anlamına gelir. Grafiklerde MSE değerleri, eğitim, doğrulama, test ve tüm veri setleri için ayrı ayrı gösterilmiş ve karşılaştırma yapılabilir hale getirilmiştir.

- Eğitim Seti**:** Eğitim setinde R değeri 0,77 ve MSE 1,8235 olarak belirtilmiş. Bu değerler, modelin eğitim sürecinde tahmin gücünün yüksek olduğunu, ancak mükemmel bir uyum sağlamadığını gösterir. Eğitimdeki R değeri ve MSE'nin orta seviyede olması, modelin eğitiminin iyi olduğunu ancak mükemmel bir uyum sağlamadığını gösterir.

- Doğrulama Seti: Doğrulama setinde R değeri 0,78 ve MSE 1,7927 olarak belirtilmiş. Bu, modelin doğrulama verileri üzerinde eğitim setine yakın bir performans sergilediğini gösterir. Eğitim ve doğrulama setindeki benzer MSE değerleri, modelin doğrulama sürecinde aşırı öğrenme (overfitting) yapmadığını ve genelleme yeteneğinin güçlü olduğunu gösterir.

- Test Seti: Test setinde R değeri 0,81 ve MSE 1,5651 olarak belirtilmiş. Test setinde en yüksek R değeri ve en düşük MSE değeri gözlemlenmiştir. Bu, modelin test verileri üzerinde daha iyi performans sergilediğini ve test sürecinde doğruluğunun arttığını gösterir.

- Tüm Veri Seti: Tüm veri seti için R değeri 0,78 ve MSE 1,7795 olarak gösterilmiştir. Tüm veriler dikkate alındığında R ve MSE değerleri, eğitim ve doğrulama setiyle uyumlu bir sonuç vermektedir. Bu, modelin genelde dengeli bir performansa sahip olduğunu gösterir.

Tüm derin öğrenme modellerinin sonucunda elde edilen değerler Çizelge 5.2 ile sunulmuş olup bu değerlerin referans alınarak karşılaştırılması Sonuç ve Öneriler bölümünde irdelenecektir.

Çizelge 5.2. Derin öğrenme modellerine ait ölçüm değerleri

Model	ANN	CNN	RNN
Doğruluk (Doğrulama)	%86,27	%75,63	%88,52
Doğruluk (Test)	%89,52	%78,09	%88,89
Doğruluk (Eğitim)	%86,55	%78,09	%87,32
MSE (Doğrulama)	1,7955	3,6751	1,7927
MSE (Test)	1,5016	3	1,5651
MSE (Eğitim)	1,9328	3,3214	1,8235
R (Doğrulama)	0,7935	0,5393	0,78
R (Test)	0,8219	0,6193	0,81
R (Eğitim)	0,7653	0,8731	0,77

6. SONUÇ VE ÖNERİLER

Yapay zekâ teknolojileri, günümüzde teknoloji bakımından en hızlı gelişen ve etkili olan teknolojilerden biridir. Bu çalışmada**,** yapay zeka ile bir S band radarına ait uçuş verilerinin analiz edilerek sınıflandırma yapabileceğine ilişkin araştırma ve uygulama yapılmıştır. Çeşitli kaynaklardan toplanan ve derlenen RKA, hız ve irtifa değerlerinden oluşan uçuş verilerinin makine ve derin öğrenme yöntemleri ile 7 farklı sınıfa ayrımı gösterilmiştir. Sınıflandırma probleminin çözülmesi amacıyla yapılan bu çalışmada, MATLAB R2023b programı ile yapay zeka tabanlı sınıflandırma algoritmalarının test setleri ile başarıları ölçülmüştür.

Makine Öğrenmesi için Classification Learner (Sınıflandırma Öğreticisi) aracı kullanılarak yapılan uygulamada ön plana çıkan modelleri başarı oranları Çizelge 5.1. ile sunulmuştur. Özellikle, Topluluk Torbalama Ağaçları (Ensemble Bagged Trees) ve Ağırlaştırılmış En Yakın Komşular (Weighted KNN) modellerinde yüksek performans gözlemlenmiştir. Bu iki modelde işlenen test verilerinde sırasıyla %95,6 ve %96,5 doğruluk oranları elde edilmiştir. Farklı sınıflardaki (hedef tiplerindeki) başarı dağılımı değişkenlik içermekte olup model performansının özetlenmesi amacıyla karışıklık matrisleriyle (Şekil-5.2) görselleştirilmiş ve tahmin edilen sınıf etiketleri ile gerçek sınıflara ait etiketlerin birbirini karşılama miktarları karşılaştırılmıştır. Her bir özelliğin (RKA, hız, irtifa) modellerin genel tahmin etkisine yönelik katkısı için PDP grafikleri incelenmiştir. Hangi özelliğin tahmin performansı için daha kritik olduğu bu grafiklerin karşılaştırılması suretiyle ele alınmış olup her sınıfa ait özellikler üzerinden nasıl bir tahmin yapıldığı detaylandırılmıştır. PDP grafikleri üzerinden, Topluluk Torbalama Ağaçları modelinin daha kararlı ve dengeli tahminler yapabileceği, büyük ve küçük cisim sınıflarında (RKA açısından) ve farklı irtifa seviyelerinde nispeten iyi sonuçlar vereceği, Ağırlaştırılmış En Yakın Komşular modelinin hız ve RKA özelliklerine duyarlı olması sayesinde daha spesifik sınıf ayrımları yapabildiği, ancak düşük irtifa veya hız seviyelerinde doğruluk tahminlerinin azalabileceği analiz edilmiştir.

Önerilen modellerin test edilmesi sonucunda, tutarlı, hızlı ve yüksek oranda doğru kararlar verdikleri görülmüştür. Böylece, makine öğrenmesine dayalı sınıflandırmaların hava hedefi sınıflandırma uygulamalarında kullanılabileceği görülmüştür.

Derin Öğrenme yöntemlerinde ise ANN, CNN ve RNN modelleri ile uygulamalar yapılmıştır. Çizelge 5.2. ile sunulan sonuçlarda, her üç modele ait doğruluk oranları ve MSE değerleri yer almaktadır. Doğrulama, eğitim ve test verilerinin doğruluk oranları açsından, ANN modelinin dengeli bir performansa sahip olduğunu, CNN modelinin diğer modellere kıyasla daha düşük doğruluğa sahip olduğu, RNN modelinde ise başarı oranlarıyla ANN modeline oldukça yakın, test ve doğrulama setlerinde biraz daha iyi bir başarı alındığı görülmüştür. Sonuç olarak, Derin Öğrenme modellerinde, doğruluk oranları açısından RNN modeli test ve doğrulama setlerinde en yüksek performansı göstermiş, zaman serileri ve ardışık verilerle çalışırken daha verimli olmasından kaynaklı sınıflandırma görevlerinde avantaj sağladığı değerlendirilmiştir.

MSE değerleri bağlamında, ANN ve RNN modellerinin hata oranları birbirine benzer ve hata oranları makul düzey sayılabilecek seviyede (düşük iken) CNN modelinde MSE değerleri daha yüksek olup gözle görülür bir hata oranı hesaplanmıştır. Genel olarak incelendiğinde, CNN modelinin yüksek MSE değerleri, doğruluktaki düşük oran ile de tutarlıdır ve bu modelin sınıflandırma görevlerinde diğer modellere göre daha zayıf performans gösterdiğini işaret etmektedir. R (Korelasyon Katsayısı) değerinin karşılaştırılmasında da ANN ve RNN modellerinin doğrusal bir ilişki kurmada oldukça iyi olduğu, her iki modelinde verilerle olan doğrusal ilişkiyi daha iyi öğrendiği, CNN modelinde ise diğer modellere kıyasla daha zayıf bir korelasyon değerinin hesaplandığı görülmüştür.

Derin Öğrenme yöntemlerinden olan RNN modelinin**,** zaman serileri veya sıralı verilerle çalışmasından dolayı, ardışık veri analizleri gerektiren sınıflandırma görevleri için iyi bir seçenek olduğu, CNN modelinin diğer iki modele göre düşük doğruluk ve yüksek MSE değerleri ile en zayıf performans göstermesinden dolayı RKA, hız ve irtifa gibi uçuş verilerine dayanan sınıflandırma görevinde tercih edilmesinin dezavantaj oluşturacağı belirlenmiştir. Görsel ve iki boyutlu veri analizinde güçlü olan CNN uygulamalarının, bir boyutlu veri oluşturan RKA, hız ve irtifa gibi sayısal değerli özelliklerin analizinde zayıf kaldığı görülmüştür. RNN'e alternatif olarak genellikle statik özelliklerin analizinde iyi performans gösteren ve daha basit bir mimariye sahip ANN'de tercih edilebilir.

Bu çalışmanın bulguları, literatür araştırmasında ele alınan önceki çalışmalarla hem benzerlikler hem de yenilikçi farklılıklar sergilemektedir. Özellikle, literatürde yaygın olarak kullanılan ANN, CNN ve RNN gibi yöntemler, çalışmamızın önerdiği model yapılandırmalarında yeniden ele alınmış ve radar hedef sınıflandırmasında

uygulanabilirlikleri detaylandırılmıştır. Bu uyum göz önüne alındığında, literatürdeki yaklaşımlarla benzer yönlerimizin, yöntemin etkinliğini destekleyen kanıtlar sunduğu söylenebilir. Literatürdeki benzer çalışmalar, genellikle veri çeşitliliği, sınıflandırma doğruluğu veya algoritmaların hız performansı üzerinde yoğunlaşmıştır. Bu bağlamda, önerilen modelin, RKA, hız ve irtifa gibi çoklu veri özelliklerini birleştirerek sınıflandırma başarısını artıran yapısıyla ve farklı sınıf kategorilerine ait sınıflandırma hedefiyle literatürdeki benzerlerinden ayrıştığı gözlemlenmiştir.

Sonuç olarak, bu tezde, yapay zekâ tekniklerinin hava hedeflerinin sınıflandırılmasındaki rolü incelenmiş ve bu alanda yapılan çalışmalar değerlendirilmiştir. Yapay zekâ, hava savunma sistemlerinin etkinliğini artırmakta ve hava sahası güvenliğine önemli katkılar sağlamaktadır. Hava sahasındaki uçakların ve diğer hava araçlarının hızlı bir şekilde sınıflandırılması, anında müdahale gerektiren durumlarda doğru karar alınması açısından önem taşımaktadır. Elde edilen sonuçlar göz önüne alındığında, yapay zekâ kullanımının radar sinyal işleme algoritmaları içerinde yaygınlaşmaya devam edeceği görülmektedir. Böylelikle, hava gözetim radarı kullanıcısı olan hava trafik kontrolörleri hava trafiğini daha güvenilir bir şekilde yönetebilecek ve hava sahası güvenliğinin sağlanabilmesi için daha fazla kaynaktan bilgi alacaktır. Sınıflandırmanın verimli olarak uygulanmasıyla çarpışma önleme, riskli durumların tespiti ve tehdit unsurlarının erken fark edilmesi gibi etkenler uçuş güvenliğini artıracaktır. Gelişen teknoloji ile birlikte, yapay zekâ tabanlı sınıflandırma sistemleri sivil ve askeri havacılık sektöründe daha geniş bir kullanım alanı bulmaya devam edecektir. İleri yıllarda, otonom hava savunma sistemlerinin geliştirilmesiyle, insan müdahalesi olmadan etkin hava sahası kontrolü mümkün kılınacaktır.

Gelecekte sınıflandırılacak hedef sayısının artırılması, eğitim ve test aşamasındaki verimliliğin iyileştirilmesi, modellerin daha karmaşık hale getirilerek yeniden yapılandırılması, [56] makalesinde bahsedilen makine öğrenmesine ait algoritmaların entegre edilmesi suretiyle oluşturulacak farklı öğrenme yöntemlerinin bir arada kullanıldığı hibrit modellerin geliştirilmesi, genelleme yeteneğinin artırılması için daha fazla veri setinin toplanması, kullanılan üç adet uçuş verisine ilave özelliklerin dahil edilmesi planlanmaktadır.

KAYNAKLAR

1. Li, X., Chen, Y. (2018). *Radar signal classification with machine learning*. IEEE Transactions on Aerospace and Electronic Systems.

2. Russell, S. J., Norvig, P. (2016). *Artificial Intelligence: A Modern Approach.* Pearson.

3. Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*. MIT Press.

4. LeCun, Y., Bengio, Y., Hinton, G. (2015). *Deep Learning*. Nature, 521(7553), 436-444.

5. Sarikaya, T. B, Yumus, D., Efe, M., Soysal, G., Kirubarajan, T. (2019). Track based UAV classification using surveillance radars. *22th International Conference on Information Fusion (FUSION)*, 1–6.

6. Liu, J., Xu, Q.Y., Chen, W.S. (2021). *Classification of birds and drone targets based on motion characteristics and random forest model using surveillance radar data*. IEEE Access, 9, 160135- 160144.

7. Roychowdhury, S., Ghosh, D. (2021). Machine learning based classification of radar signatures of drones. *2nd International Conference on Range Technology (ICORT)*, 1-5.

8. Phanindra, B. R., Pralhad, R. N., Raj, A. A. B. (2020). Machine learning based classification of ducted and non-ducted propeller type quadcopter. *6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 1296-1301.

9. Nakamura, R., Hadama, H. (2017). Characteristics of ultra-wideband radar echoes from a drone. *IEICE Communications Express (ComEX)*, 6 (9), 530-534.

10. Aydemir, M., Göse, E. (2012). Radar cross section identification of air targets using the Cosine transform and neural networks. *Recent Patents on Engineering*, 6(1), 65-69.

11. Kaya, E., Kaplan, G. B. (2021). Neural network based drone recognition techniques with non-coherent S-band radar. *IEEE Radar Conference (RadarConf)*,1-6.

12. Chen, J., Li, Y., Liu, D., Yang, Y. (2022). Identification of typical partial discharge defects of distribution system equipment based on classification learner. *The 7th Asia Conference on Power and Electrical Engineering (ACPEE)*, 2025-2029.

13. Bentes, C., Velotto, D., Tings, B. (2018). Ship classification in TerraSAR-X images with convolutional neural networks. *IEEE Journal of Oceanic Engineering*, 43(1), 258-266.

14. Gomez, S., Johnson, A., Babu, P. R. (2023). Classification of radar targets based on micro-doppler features using high frequency high resolution radar signatures. *2023 International Conference on Network, Multimedia and Information Technology (NMITCON)*, IEEE.

15. Cai, X., Giallorenzo, M., Sarabandi, K. (2021). Machine learning-based target classification for MMW radar in autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 6(4), 678-689.

16. Ardon, G., Simko, O., Novoselsky, A. (2020). Aerial radar target classification using artificial neural networks. *The 9th International Conference on Pattern Recognition Applications and Methods-ICPRAM*, 136-141.

17. Sajjan, S. S., Bhumika, C., Balamati, C., Raveendranath. N. (2019). Machine-learning for classification of naval targets. *MTT-S International Microwave and RF Conference (IMARC),* 1-4.

18. Kim, A. R., Kim, H. S., Kang, C. H., Kim, S. Y. (2023). Design of the 1D CNN–GRU network based on the RCS for classification of multiclass missiles. *Remote Sensing*, 15(3), 577.

19. Holt, D., Guo, W., Sun, M., Panagiotakopoulos, D., Warston, H. (2024). Deep learning for radar classification. *International Conference on Unmanned Aircraft Systems (ICUAS),* 1208–1215.

20. Kumawat, H. C., Chakraborty, M., Bazil Raj, A. A., Dhavale, S. V. (2022). DIAT-μSAT: Small aerial targets'micro-doppler signatures and their classification using CNN. *IEEE Geoscience and Remote Sensing Letters,*19, 1-5.

21. Karlsson, A., Jansson, M., Hämäläinen, M. (2022). Model-aided drone classification using convolutional neural networks. *IEEE Radar Conference (RadarConf22*), 1-6.

22. Jane, F. (1999). *Jane's all the world's aircraft*. McGraw-Hill.

23. Dokter, A. M., Liechti, F., Stark, H., Delobbe, L., Tabary, P., Holleman, I. (2011). Bird migration flight altitudes studied by a network of operational weather radars. *Journal of the Royal Society Interface,* 8, 30-43.

24. May, R., Steinheim, Y., Kvaløy, P., Vang, R., Hanssen, F. (2017). Performance test and verification of an off-the-shelf automated avian radar tracking system. *Ecology and Evolution,* 7, 5930-5938.

25. Moon, J. R. (2002). Effects of birds on radar tracking systems. *Radar Conference*, 300-304.

26. Nilsson, C., Dokter, A., M., Schmid, B. (2018). Field validation of radar systems for monitoring bird migration. *Journal of Applied Ecology*, 55, 2552–2564.

27. Yang, P., Ting, G., Qiang, D., Bifeng, S. (2011). Qualitative assessment of the RCS reduction benefits on detection probability of an armed helicopter. *Applied Mechanics and Materials*, 55 (57), 1535-1540.

28. Irwin, D., Queen., F. D. (1965). *Dynamic measurement of radar cross sections*, 53(8), 954-961

29. Nakamura, R., Hadama, H. (2015). Characteristics of ultra-wideband radar echoes from a drone. *IEICE Communications Express (ComEX),* 4 (6), 130-184.

30. *DJI Mini 4 Pro Kullanım Klavuzu*, v. 1. (2023).

31. *Phantom 3 Standard User Manuel*, v. 4. (2017).

32. Coşkun, M. (2021). *Hava savunma sistemlerinde bulanık mantık tabanlı tehdit değerlendirmesi ve derecelendirmesi*. Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Konya, 2.

33. Chung, S. M., Tuan, S. C. (2020). Radar cross section simulation of XQ-58 Valkyrie like CAD model. *2020 International Workshop on Electromagnetics: Applications and Student Innovation Competition (iWEM)*, 1-2.

34. Steinheim, Y. (2016). *Radar Cross Section measurement of model aircraft used as avian radar test target*. The Norwegian Institute for Nature Research.

35. Yang, C., Yan, W., Zhao, Y., Geng, Hou, S., Chen, J. (2020). RCS optimization analysis method for sea-skimming unmanned aerial vehicle based on back propagation neural network algorithm. *The Applied Computational Electromagnetics Society (ACES) Journal*, 35.

36. Zikidis, K., Skondras, A., Tokas, C. (2014). Low observable principles, stealth aircraft and anti-stealth technologies. *Journal of Computations & Modelling*, 4(1), 129-165.

37. Yue, K., Liu, W., Li, G., Ji, J., Yu, D. (2015). Numerical simulation of RCS for carrier electronic warfare airplanes. *Chinese Journal of Aeronautics*, 28(2), 545-555.

38. Wilson, J. D. (1972). Probability of detecting aircraft targets. *IEEE Transactions on Aerospace and Electronic Systems*, 8(6), 757-761.

39. Ashraf, R., Tabassum, S. T., Hossam, M. H. (2018. Analytical study of bi-static radar cross section with a comparison at S band and X band of F-117 Nighthawk stealth aircraft. *4th International Conference on Electrical Engineering and Information, Communication Technology (iCEEiCT)*, 406-410.

40. Wind, H. J., Kloke, K. H., Böniger, U., Pratisto, H. (2015). Comparison of the recorded RCS and spectra of three different wind turbines on L and S band. *2015 IEEE Radar Conference*, 266-271.

41. Greving, G., Malkomes, M. (2010). Weather radar and wind turbines-an update of the theoretical and numerical analysis of effects. *ERAD 2010-The Sıxth European Conference On Radar In Meteorology And Hydrology*.

42. Tumolo, R. M., Immediata, S., D'Urso, M. (2013). Wind farm in radar systems: Modeling, analysis, simulations and possible countermeasures. *2013 IEEE International Symposium on Phased Array Systems and Technology*, 180-184.

43. McCarthy, J. (1956). *Dartmouth Conference Proposal*, Dartmouth Conference.

44. Akın, E., Şahin, M. E. (2024). A study on deep learning and artificial neural network models. *EMO Bilimsel Dergi*, 14(1), 27-38.

45. Samuel, A. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3, 210-229.

46. İnternet: Turhost. (2024). URL: https://www.turhost.com/blog/makine-ogrenmesi-machine-learning-nedir/, Son Erişim Tarihi: 16.10.2024.

47. Turing, A. M. (1950). *Computing machinery and intelligence*, 59(236), 433-460.

48. Hussain, M., Kanwal Wajid, S., Elzaart, A., Berbar, M. (2011). A comparison of SVM kernel functions for breast cancer detection. *8th International Conference Computer Graphics, Imaging and Visualization*, 145-150.

49. Karal, O. (2020). Performance comparison of different kernel functions in SVM for different k value in k-fold cross-validation. *Innovations in Intelligent Systems and Applications Conference (ASYU)*, 1-5.

50. Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1), 81-106.

51. Zhang, Z. (2016). Introduction to machine learning: K-nearest neighbors. *Annals of Translational Medicine*, 4(11), 218.

52. İnternet: Sönmez, D. (2024) Geri Yayılım Algoritması'na Matematiksel Yaklaşım, URL: http://www.derinogrenme.com/2018/06/28/geri-yayilim algoritmasina - matematiksel- yaklasim/, Son Erişim Tarihi: 14.09.2024.

53. Hinton, G. E., Osindero, S., Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7), 1527-1554.

54. Khan, S., Rahmani, H., Shah, S. A. A., Bennamoun, M. (2018). A guide to convolutional neural networks for computer vision. *Synthesis Lectures on Computer Vision*, 8(1), 1-207.

55. Gavcar, E., Metin, H. M. (2021). *Hisse senedi değerlerinin makine öğrenimi (derin öğrenme) ile tahmini*, 10 (2), 1–11.

56. Başkaya, K., Kurt, E. (2024). Aerial target identification based on classification learner. *4th Interdisciplinary Conference on Electrics and Computer (INTCEC).* doi:10.1109/INTCEC61833.2024.10603187.

EKLER

EK-1. Sınıflandırma öğreticisine ait kod

```
function [trainedClassifier, validationAccuracy] = trainClassifier(trainingData,
responseData)

% [trainedClassifier, validationAccuracy] = trainClassifier(trainingData,
% responseData)
% Returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
%  Input:
%
%      trainingData: A matrix with the same number of columns and data type
%       as the matrix imported into the app.
%
%      responseData: A vector with the same data type as the vector
%       imported into the app. The length of responseData and the number of
%       rows of trainingData must be equal.
%
%
%  Output:
%
%      trainedClassifier: A struct containing the trained classifier. The
%       struct contains various fields with information about the trained
%       classifier.
%
%      trainedClassifier.predictFcn: A function to make predictions on new
%       data.
%
%      validationAccuracy: A double representing the validation accuracy as
%       a percentage. In the app, the Models pane displays the validation
%       accuracy for each model.
%
% Use the code to train the model with new data. To retrain your
% classifier, call the function from the command line with your original
% data or new data as the input arguments trainingData and responseData.
%
% For example, to retrain a classifier trained with the original data set T
% and response Y, enter:
%   [trainedClassifier, validationAccuracy] = trainClassifier(T, Y)
%
% To make predictions with the returned 'trainedClassifier' on new data T2,
% use
%   [yfit,scores] = trainedClassifier.predictFcn(T2)
%
% T2 must be a matrix containing only the predictor columns used for
% training. For details, enter:
%   trainedClassifier.HowToPredict
%
% Auto-generated by MATLAB on 10-Nov-2024 22:38:56
% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
% Convert input to table

inputTable = array2table(trainingData, 'VariableNames', {'column_1', 'column_2',
'column_3'});
```

EK-1. (devam) Sınıflandırma öğreticisine ait kod

```
predictorNames = {'column_1', 'column_2', 'column_3'};
predictors = inputTable(:, predictorNames);
response = responseData;
isCategoricalPredictor = [false, false, false];
classNames = [1; 2; 3; 4; 5; 6; 7];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.

classificationKNN = fitcknn(...
    predictors, ...
    response, ...
    'Distance', 'Euclidean', ...
    'Exponent', [], ...
    'NumNeighbors', 10, ...
    'DistanceWeight', 'SquaredInverse', ...
    'Standardize', true, ...
    'ClassNames', classNames);

% Create the result struct with predict function
predictorExtractionFcn = @(x) array2table(x, 'VariableNames', predictorNames);
knnPredictFcn = @(x) predict(classificationKNN, x);
trainedClassifier.predictFcn = @(x) knnPredictFcn(predictorExtractionFcn(x));
% Add additional fields to the result struct

trainedClassifier.ClassificationKNN = classificationKNN;
trainedClassifier.About = 'This struct is a trained model exported from
Classification Learner R2023b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new predictor
column matrix, X, use: \n  [yfit,scores] = c.predictFcn(X) \nreplacing ''c'' with
the name of the variable that is this struct, e.g. ''trainedModel''. \n \nX must
contain exactly 3 columns because this model was trained using 3 predictors. \nX
must contain only predictor columns in exactly the same order and format as your
training \ndata. Do not include the response column or any columns you did not
import into the app. \n \nFor more information, see <a
href="matlab:helpview(fullfile(docroot, ''stats'', ''stats.map''),
''appclassification_exportmodeltoworkspace'')">How to predict using an exported
model</a>.');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
% Convert input to table

inputTable = array2table(trainingData, 'VariableNames', {'column_1', 'column_2',
'column_3'});
predictorNames = {'column_1', 'column_2', 'column_3'};
predictors = inputTable(:, predictorNames);
response = responseData;
isCategoricalPredictor = [false, false, false];
classNames = [1; 2; 3; 4; 5; 6; 7];
% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationKNN, 'KFold', 5);
% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);
% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');
```

EK-2. ANN modeline ait kod

```
% Veriyi yükle
load('ysaveri1.mat');
% Verilerin boyutlarını kontrol et
XTrain = trainysaveri;  % Eğitim için giriş verileri
YTrain = trainysasonuc; % Eğitim için hedef etiketler
XTest = testysaveri;    % Test için giriş verileri
YTest = testysasonuc;    % Test için hedef etiketler
% Verileri uygun formata getir
XTrain = XTrain';   % Eğitim verisi (Özellik sayısı, Örnek sayısı) şeklinde
transpoze edildi
XTest = XTest';   % Test verisi (Özellik sayısı, Örnek sayısı) şeklinde transpoze
edildi
% Hedef etiketleri kategorik hale getir
YTrain = categorical(YTrain);
YTest = categorical(YTest);
% Eğitim verisinin boyutlarını kontrol et
disp(['XTrain boyutu: ', num2str(size(XTrain))]);
disp(['YTrain boyutu: ', num2str(size(YTrain))]);
% Check if dimensions match before splitting
assert(size(XTrain,   2)   ==   numel(YTrain),   'XTrain   ve   YTrain   boyutları
eşleşmiyor.');
% Eğitim verisini %80 eğitim, %20 doğrulama olarak ayırma
cv = cvpartition(numel(YTrain), 'HoldOut', 0.2); % %20 doğrulama için ayır
idx = cv.test; % Test indekslerini al

% Doğrulama setini oluşturma
XValidation = XTrain(:, idx);    % Doğrulama verisi
YValidation = YTrain(idx);        % Doğrulama etiketleri

% Kalan eğitim verisini ayırma
XTrain = XTrain(:, ~idx);        % Eğitim verisi
YTrain = YTrain(~idx);            % Eğitim etiketleri

% Eğitim ve doğrulama verisinin boyutlarını kontrol et
disp(['Yeni XTrain boyutu: ', num2str(size(XTrain))]);
disp(['Yeni YTrain boyutu: ', num2str(size(YTrain))]);
disp(['XValidation boyutu: ', num2str(size(XValidation))]);
disp(['YValidation boyutu: ', num2str(size(YValidation))]);
% Neural Network Modeli Tanımlama

input_size = size(XTrain, 1);  % Giriş veri boyutu (özellik sayısı)
num_classes = numel(categories(YTrain));  % Sınıf sayısı
layers = [
    featureInputLayer(input_size, 'Normalization', 'zscore')   % Giriş katmanı
(normalizasyon ile)
    fullyConnectedLayer(128)                                  % 128 birimlik tam
bağlantı katmanı
    reluLayer                                                      % Aktivasyon
fonksiyonu (ReLU)
    fullyConnectedLayer(64)                                   % 64 birimlik tam
bağlantı katmanı
    reluLayer                                                      % Aktivasyon
fonksiyonu (ReLU)
    fullyConnectedLayer(num_classes)                            % Çıkış katmanı
(sınıf sayısı kadar)
    softmaxLayer                                              % Softmax katmanı
classificationLayer];                                       % Sınıflandırma katmanı
```

EK-2. (devam) ANN modeline ait kod

```
% Eğitim opsiyonlarını belirleme
options = trainingOptions('adam', ...
    'MaxEpochs', 30, ...        % Epoch sayısı
    'MiniBatchSize', 32, ...    % Mini batch boyutu
    'ValidationData', {XValidation', YValidation}, ... % Doğrulama verisi
    'Verbose', false, ...
    'Plots', 'training-progress');  % Eğitim ilerlemesini gösteren grafik
% Modeli eğitme
net = trainNetwork(XTrain', YTrain, layers, options);  % Eğitim için transpoze!
%% Eğitim Verisi için Tahmin ve Confusion Matrix
YPredTrain = classify(net, XTrain');  % Eğitim setinde tahmin yap
figure;
cmTrain = confusionchart(YTrain, YPredTrain);   % Eğitim seti için karışıklık
matrisi
title('Confusion Matrix - Eğitim Seti');
% Başarı oranlarını hesaplama
train_class_accuracies = zeros(numel(categories(YTrain)), 1);
classes = categories(YTrain);
for i = 1:numel(classes)
    class_idx_train = (YTrain == classes{i});
    class_accuracy_train       =       sum(YPredTrain(class_idx_train)       ==
YTrain(class_idx_train)) / sum(class_idx_train);
    train_class_accuracies(i) = class_accuracy_train;
end
% Başarı oranlarını ekleme

axes('Position', get(cmTrain.Parent, 'Position'), 'Color', 'none', 'XColor',
'none', 'YColor', 'none');
for i = 1:numel(classes)
    text(i, max(cmTrain.NormalizedValues(:)) + 0.02, ...
         sprintf('%.2f', train_class_accuracies(i)), ...
         'Color', 'r', 'FontSize', 12, ...
         'HorizontalAlignment', 'center');
end

%% Test Verisi için Tahmin ve Confusion Matrix
YPredTest = classify(net, XTest');  % Test setinde tahmin yap
figure;
cmTest = confusionchart(YTest, YPredTest);  % Test seti için karışıklık matrisi
title('Confusion Matrix - Test Seti');
% Başarı oranlarını hesaplama
test_class_accuracies = zeros(numel(classes), 1);
for i = 1:numel(classes)
    class_idx_test = (YTest == classes{i});
    class_accuracy_test        =        sum(YPredTest(class_idx_test)        ==
YTest(class_idx_test)) / sum(class_idx_test);
    test_class_accuracies(i) = class_accuracy_test;
end
% Başarı oranlarını ekleme

axes('Position', get(cmTest.Parent, 'Position'), 'Color', 'none', 'XColor',
'none', 'YColor', 'none');
for i = 1:numel(classes)
    text(i, max(cmTest.NormalizedValues(:)) + 0.02, ...
sprintf('%.2f', test_class_accuracies(i)), ...
```

EK-2. (devam) ANN modeline ait kod

```
'Color', 'r', 'FontSize', 12, ...

'HorizontalAlignment', 'center');
end

%% ROC Analizi - Eğitim verisi için
scoresTrain = predict(net, XTrain');
figure;
hold on;
for i = 1:numel(classes)
    classLabel = classes{i};
    [XTrain_ROC, YTrain_ROC, ~, AUC_Train] = perfcurve(YTrain == classLabel,
scoresTrain(:, i), true);
    plot(XTrain_ROC, YTrain_ROC, 'DisplayName', ['Class ' classLabel ' (AUC: '
num2str(AUC_Train) ')']);
end
xlabel('False Positive Rate');
ylabel('True Positive Rate');
title('ROC Curve - Eğitim Seti (Tüm Sınıflar)');
legend('show');
hold off;

%% Tahminler için Doğrulama Verisi
YPredValidation = classify(net, XValidation');  % Doğrulama setinde tahmin yap
% Başarı oranlarını hesaplama

validation_class_accuracies = zeros(numel(categories(YValidation)), 1);
for i = 1:numel(classes)
    class_idx_val = (YValidation == classes{i});
    class_accuracy_val       =       sum(YPredValidation(class_idx_val)       ==
YValidation(class_idx_val)) / sum(class_idx_val);
    validation_class_accuracies(i) = class_accuracy_val;
end
% ROC Analizi - Doğrulama verisi için

scoresValidation = predict(net, XValidation');
figure;
hold on;
for i = 1:numel(classes)
    classLabel = classes{i};
    [XVal_ROC,   YVal_ROC,   ~,   AUC_Validation]   =   perfcurve(YValidation   ==
classLabel, scoresValidation(:, i), true);
    plot(XVal_ROC, YVal_ROC, 'DisplayName', ['Class ' classLabel ' (AUC: '
num2str(AUC_Validation) ')']);
end
xlabel('False Positive Rate');
ylabel('True Positive Rate');
title('ROC Curve - Doğrulama Seti (Tüm Sınıflar)');
legend('show');
hold off;
mse_train = mean((double(YPredTrain) - double(YTrain)).^2);
mse_test = mean((double(YPredTest) - double(YTest)).^2);
mse_validation = mean((double(YPredValidation) - double(YValidation)).^2); % MSE
for validation data
disp(['Eğitim verisi MSE değeri: ', num2str(mse_train)]);
disp(['Test verisi MSE değeri: ', num2str(mse_test)]);
```

EK-2. (devam) ANN modeline ait kod

```
disp(['Doğrulama verisi MSE değeri: ', num2str(mse_validation)]);
%% Eğitim, Test ve Doğrulama Veri Seti için Regresyon Analizi
YPredTotal = [YPredTrain; YPredTest; YPredValidation];  % Toplam tahminler
YTotal = [YTrain; YTest; YValidation];                  % Toplam gerçek değerler
figure;
plotregression(double(YTrain), double(YPredTrain), 'Eğitim Seti', ...
               double(YTest), double(YPredTest), 'Test Seti', ...
               double(YValidation),  double(YPredValidation),  'Doğrulama  Seti',
... % Validation data
               double(YTotal), double(YPredTotal), 'Tüm Veri Seti');
title('Regression Analysis - Eğitim, Test ve Doğrulama Seti');

%% Her sınıfın başarı oranını gösteren çubuk grafik

figure;
bar(train_class_accuracies);
set(gca, 'xticklabel', classes);
title('Eğitim Seti - Her Sınıf İçin Doğruluk Oranı');
xlabel('Sınıflar');
ylabel('Doğruluk Oranı');
ylim([0 1]);
figure;
bar(test_class_accuracies);
set(gca, 'xticklabel', classes);
title('Test Seti - Her Sınıf İçin Doğruluk Oranı');
xlabel('Sınıflar');
ylabel('Doğruluk Oranı');
ylim([0 1]);
figure;
bar(validation_class_accuracies);
set(gca, 'xticklabel', classes);
title('Doğrulama Seti - Her Sınıf İçin Doğruluk Oranı');
xlabel('Sınıflar');
ylabel('Doğruluk Oranı');
ylim([0 1]);
```

EK-3. CNN modeline ait kod

```
% Veriyi yükle
load('ysaveri1.mat');
% Verilerin boyutlarını kontrol et
XTrain = trainysaveri;  % Eğitim için giriş verileri
YTrain = trainysasonuc; % Eğitim için hedef etiketler
XTest = testysaveri;    % Test için giriş verileri
YTest = testysasonuc;   % Test için hedef etiketler

% Verileri uygun formata getir
XTrain = XTrain';   % Eğitim verisi (Özellik sayısı, Örnek sayısı) şeklinde
transpoze edildi
XTest = XTest';    % Test verisi (Özellik sayısı, Örnek sayısı) şeklinde transpoze
edildi
% Hedef etiketleri kategorik hale getir

YTrain = categorical(YTrain);
YTest = categorical(YTest);
% Veri kümesini eğitim ve doğrulama setlerine ayır
```

EK-3. (devam) CNN modeline ait kod

```
cv = cvpartition(YTrain, 'HoldOut', 0.2); % %20 doğrulama seti
idx = cv.test;
% Eğitim ve doğrulama verilerini oluştur
XValidation = XTrain(:, idx);  % Doğrulama verisi
YValidation = YTrain(idx);      % Doğrulama etiketleri
XTrain(:, idx) = [];            % Eğitim verisinden doğrulama örneklerini çıkar
YTrain(idx) = [];                % Eğitim etiketlerinden doğrulama etiketlerini
çıkar
% CNN Modeli için uygun formata dönüştürme (N x H x W x C formatı)
num_samples_train = size(XTrain, 2); % Eğitim örnek sayısı
num_samples_test = size(XTest, 2);   % Test örnek sayısı
num_samples_validation = size(XValidation, 2); % Doğrulama örnek sayısı
% Verileri 3D diziye dönüştürme (Özellik sayısı x Yükseklik x Genişlik x Örnek
sayısı)
XTrain = reshape(XTrain, size(XTrain, 1), 1, 1, num_samples_train); % (Özellik
sayısı, 1, 1, Örnek sayısı)
XTest = reshape(XTest, size(XTest, 1), 1, 1, num_samples_test);      % (Özellik
sayısı, 1, 1, Örnek sayısı)
XValidation    =    reshape(XValidation,    size(XValidation,    1),    1,    1,
num_samples_validation); % (Özellik sayısı, 1, 1, Doğrulama örnek sayısı)
4% CNN Modeli Tanımlama
input_size = size(XTrain, 1);  % Giriş veri boyutu (özellik sayısı)
num_classes = numel(categories(YTrain));  % Sınıf sayısı
layers = [
    imageInputLayer([input_size 1 1])  % Giriş katmanı (yükseklik ve genişlik 1)
    convolution2dLayer([input_size  1],  16,  'Padding',  'same')  %  Konvolüsyon
katmanı
    reluLayer                                % ReLU aktivasyon
    maxPooling2dLayer([1 1])                % Max pooling
    fullyConnectedLayer(num_classes)         % Tam bağlantılı katman
    softmaxLayer                             % Softmax katmanı
    classificationLayer                      % Sınıflandırma katmanı
];

% Eğitim opsiyonlarını belirleme
options = trainingOptions('adam', ...
    'MaxEpochs', 30, ...        % Epoch sayısı
    'MiniBatchSize', 32, ...    % Mini batch boyutu
    'ValidationData', {XValidation, YValidation}, ... % Doğrulama verisi
    'Shuffle', 'every-epoch', ... % Her epoch'da veriyi karıştır
    'Verbose', false, ...
    'Plots', 'training-progress');  % Eğitim ilerlemesini gösteren grafik
% Modeli eğitme
net = trainNetwork(XTrain, YTrain, layers, options);
%% Eğitim ve Test Seti için Tahmin ve Confusion Matrix
YPredTrain = classify(net, XTrain);  % Eğitim setinde tahmin yap
YPredTest = classify(net, XTest);    % Test setinde tahmin yap
YPredValidation = classify(net, XValidation); % Doğrulama setinde tahmin yap
% Eğitim, Test ve Doğrulama setleri için karmaşıklık matrisi
figure;
confusionchart(YTrain, YPredTrain);
title('Confusion Matrix - Eğitim Seti');
figure;
confusionchart(YTest, YPredTest);
title('Confusion Matrix - Test Seti');
```

EK-3. (devam) CNN modeline ait kod

```
figure;
confusionchart(YValidation, YPredValidation);
title('Confusion Matrix - Doğrulama Seti');
%% Mean Squared Error (MSE) Hesaplama
mse_train = mean((double(YPredTrain) - double(YTrain)).^2);
mse_test = mean((double(YPredTest) - double(YTest)).^2);
mse_validation = mean((double(YPredValidation) - double(YValidation)).^2);
disp(['Eğitim verisi MSE değeri: ', num2str(mse_train)]);
disp(['Test verisi MSE değeri: ', num2str(mse_test)]);
disp(['Doğrulama verisi MSE değeri: ', num2str(mse_validation)]);
%% Test Seti İçin Başarı Oranı (Accuracy)
test_accuracy = sum(YPredTest == YTest) / numel(YTest) * 100;
disp(['Test Seti Başarı Oranı: ', num2str(test_accuracy), '%']);
%% Her sınıfın başarı oranını gösteren çubuk grafik (Test verisi)
test_class_accuracies = zeros(numel(categories(YTest)), 1);
classes = categories(YTest);
disp('--- Test Verisi Sınıf Bazında Doğruluk ---');
for i = 1:numel(classes)
    class = classes{i};
    % Her sınıf için doğruluk oranını hesapla
    class_idx_test = (YTest == class);
    class_accuracy_test        =        sum(YPredTest(class_idx_test)        ==
YTest(class_idx_test)) / sum(class_idx_test);
    test_class_accuracies(i) = class_accuracy_test;
    disp(['Sınıf ', class, ' doğruluk (Test): ', num2str(class_accuracy_test *
100), '%']);
end

% Test seti için doğruluk oranları

figure;
bar(test_class_accuracies);
set(gca, 'xticklabel', classes);  % X eksenine sınıf isimlerini ekle
title('Test Seti - Her Sınıf İçin Doğruluk Oranı');
xlabel('Sınıflar');
ylabel('Doğruluk Oranı');
ylim([0 1]);

% Eğitim, Test ve Doğrulama Verisi için Regresyon Analizi
YPredTotal = [YPredTrain; YPredTest; YPredValidation];
YTotal = [YTrain; YTest; YValidation];
figure;
plotregression(double(YTrain), double(YPredTrain), 'Eğitim Seti', ...
               double(YTest), double(YPredTest), 'Test Seti', ...
               double(YValidation),  double(YPredValidation),  'Doğrulama  Seti',
...
               double(YTotal), double(YPredTotal), 'Tüm Veri Seti');
title('Regression Analysis - Eğitim, Test, Doğrulama ve Tüm Veri Seti');
```

EK-4. RNN modeline ait kod

```
% Veriyi yükle
load('ysaveri1.mat');
% Ön işleme
XTrain = num2cell(trainysaveri', 1);  % Eğitim verisini hücre dizisine dönüştür
(sıralı veri için)
```

EK-4. (devam) RNN modeline ait kod

```
YTrain = categorical(trainysasonuc);  % Hedef etiketleri kategorik veriye dönüştür
XTest = num2cell(testysaveri', 1);    % Test verisini hücre dizisine dönüştür
YTest = categorical(testysasonuc);      % Test etiketlerini kategorik veriye
dönüştür
% Eğitim verisini %80 eğitim, %20 doğrulama olarak ayır
cv = cvpartition(YTrain, 'HoldOut', 0.2);
idx = cv.test;  % Test verisini elde et
% Doğrulama setini oluştur
XValidation = XTrain(idx);    % Doğrulama verisi
YValidation = YTrain(idx);    % Doğrulama etiketleri
% Eğitim setinden doğrulama verisini çıkar
XTrain(idx) = [];              % Eğitim verisinden doğrulama verisini çıkar
YTrain(idx) = [];                % Eğitim etiketlerinden doğrulama etiketlerini
çıkar
% RNN mimarisini tanımla
inputSize = size(XTrain{1}, 1);  % Girdi boyutu (özellik sayısı)
numClasses = numel(categories(YTrain));  % Hedef sınıf sayısı
layers = [
    sequenceInputLayer(inputSize)  % Girdi katmanı
    lstmLayer(100, 'OutputMode', 'last')  % LSTM katmanı (100 gizli birim)
    fullyConnectedLayer(numClasses)  % Tam bağlı katman
    softmaxLayer  % Sınıflandırma için softmax katmanı
    classificationLayer];  % Kategorik sınıflandırma katmanı
% Eğitim seçeneklerini belirle
options = trainingOptions('adam', ...
    'MaxEpochs', 30, ...  % Maksimum epoch sayısı
    'MiniBatchSize', 32, ...  % Mini batch boyutu
    'ValidationData', {XValidation, YValidation}, ... % Doğrulama verisini ekle
    'Verbose', false, ...
    'Plots', 'training-progress', ...
    'ExecutionEnvironment', 'auto', ...  % İstemci ortamını otomatik olarak seç
    'ValidationPatience', 5);  % Erken durdurma için doğrulama sabrı
% RNN modelini eğit
net = trainNetwork(XTrain, YTrain, layers, options);
% Tahmin ve Değerlendirme
% Eğitim verisi üzerinde tahmin yap
YPredTrain = classify(net, XTrain);
YPredProbTrain = predict(net, XTrain);  % ROC için olasılıkları elde et
% Doğrulama verisi üzerinde tahmin yap
YPredValidation = classify(net, XValidation);
YPredProbValidation = predict(net, XValidation);  % ROC için olasılıkları elde
et
% Test verisi üzerinde tahmin yap
YPredTest = classify(net, XTest);
YPredProbTest = predict(net, XTest);  % ROC için olasılıkları elde et
% Eğitim seti için karışıklık matrisi

figure;
cmTrain = confusionchart(YTrain, YPredTrain);
cmTrain.RowSummary = 'row-normalized';
title('Confusion Matrix - Training Set');
% Doğrulama seti için karışıklık matrisi

figure;
cmValidation = confusionchart(YValidation, YPredValidation);
cmValidation.RowSummary = 'row-normalized';
title('Confusion Matrix - Validation Set');
```

EK-4. (devam) RNN modeline ait kod

```
% Test seti için karışıklık matrisi
figure;
cmTest = confusionchart(YTest, YPredTest);
cmTest.RowSummary = 'row-normalized';
title('Confusion Matrix - Test Set');
% Eğitim verisi için sınıf doğruluklarını hesapla
train_class_accuracies = zeros(numClasses, 1);
disp('--- Eğitim Verisi Sınıf Bazında Doğruluk ---');
trainCategories = categories(YTrain);
for i = 1:numClasses
    class = trainCategories{i};
    class_idx = (YTrain == class);
    class_accuracy = sum(YPredTrain(class_idx) == YTrain(class_idx)) /
sum(class_idx);
    train_class_accuracies(i) = class_accuracy;
    disp(['Sınıf ', class, ' için doğruluk oranı (Eğitim): ',
num2str(class_accuracy)]);
end
% Doğrulama verisi için sınıf doğruluklarını hesapla

validation_class_accuracies = zeros(numClasses, 1);
disp('--- Doğrulama Verisi Sınıf Bazında Doğruluk ---');
validationCategories = categories(YValidation);
for i = 1:numClasses
    class = validationCategories{i};
    class_idx = (YValidation == class);
    class_accuracy = sum(YPredValidation(class_idx) == YValidation(class_idx)) /
sum(class_idx);
    validation_class_accuracies(i) = class_accuracy;
    disp(['Sınıf ', class, ' için doğruluk oranı (Doğrulama): ',
num2str(class_accuracy)]);
end

% Test verisi için sınıf doğruluklarını hesapla

test_class_accuracies = zeros(numClasses, 1);
disp('--- Test Verisi Sınıf Bazında Doğruluk ---');
testCategories = categories(YTest);
for i = 1:numClasses
    class = testCategories{i};
    class_idx = (YTest == class);
    class_accuracy = sum(YPredTest(class_idx) == YTest(class_idx)) /
sum(class_idx);
    test_class_accuracies(i) = class_accuracy;
    disp(['Sınıf ', class, ' için doğruluk oranı (Test): ',
num2str(class_accuracy)]);
end
% Eğitim, doğrulama ve test verisi için sınıf doğruluklarını bar grafiği ile
görselleştir

figure;
subplot(3,1,1);
bar(train_class_accuracies);
set(gca, 'xticklabel', trainCategories);
title('Sınıf Bazında Başarı Oranı (Eğitim Seti)');
xlabel('Sınıflar');
ylabel('Doğruluk Oranı');
```

EK-4. (devam) RNN modeline ait kod

```
ylim([0 1]);
subplot(3,1,2);
bar(validation_class_accuracies);
set(gca, 'xticklabel', validationCategories);
title('Sınıf Bazında Başarı Oranı (Doğrulama Seti)');
xlabel('Sınıflar');
ylabel('Doğruluk Oranı');
ylim([0 1]);
subplot(3,1,3);
bar(test_class_accuracies);
set(gca, 'xticklabel', testCategories);
title('Sınıf Bazında Başarı Oranı (Test Seti)');
xlabel('Sınıflar');
ylabel('Doğruluk Oranı');
ylim([0 1]);
% Regresyon Analizi ve MSE Hesaplaması

allData = [YTrain; YValidation; YTest];
allPredictions = [YPredTrain; YPredValidation; YPredTest];
allPredictionsNumeric = double(allPredictions);
% Regresyon sonuçlarını görselleştir
figure;
hold on;
% Eğitim Seti Regresyon Grafiği

subplot(2, 2, 1);
scatter(double(YTrain), double(YPredTrain), 'b', 'DisplayName', 'Eğitim Seti');
xlabel('Gerçek Değerler (Eğitim)');
ylabel('Tahminler (Eğitim)');
title('Eğitim Seti: Gerçek vs Tahmin Değerleri');
grid on;
hold off;
% Doğrulama Seti Regresyon Grafiği

subplot(2, 2, 2);
scatter(double(YValidation),   double(YPredValidation),   'g',   'DisplayName',
'Doğrulama Seti');
xlabel('Gerçek Değerler (Doğrulama)');
ylabel('Tahminler (Doğrulama)');
title('Doğrulama Seti: Gerçek vs Tahmin Değerleri');
grid on;
% Test Seti Regresyon Grafiği

subplot(2, 2, 3);
scatter(double(YTest), double(YPredTest), 'r', 'DisplayName', 'Test Seti');
xlabel('Gerçek Değerler (Test)');
ylabel('Tahminler (Test)');
title('Test Seti: Gerçek vs Tahmin Değerleri');
grid on;
% Ortalama Kare Hatası (MSE) Hesaplaması

mseTrain = mean((double(YTrain) - double(YPredTrain)).^2);
mseValidation = mean((double(YValidation) - double(YPredValidation)).^2);
mseTest = mean((double(YTest) - double(YPredTest)).^2);
disp(['Eğitim Seti MSE: ', num2str(mseTrain)]);
disp(['Doğrulama Seti MSE: ', num2str(mseValidation)]);
disp(['Test Seti MSE: ', num2str(mseTest)]);
```

EK-5. Makale yayını

4. Interdisciplinary Conference on Electrics and Computer (INTCEC 2024)
11-13 June 2024, Chicago-USA

Aerial Target Identification Based on Classification Learner

Kadir Başkaya
Institute of Sciences, Gazi University,
TR06500 Beşevler, Ankara, Türkiye
kadir.baskaya@gazi.edu.tr
ORCID: 0009-0008-2030-6804

Erol Kurt
Dep. Elec. Electr. Engin., Tech. Faculty, Gazi University,
TR06500 Beşevler, Ankara, Türkiye
ekurt52tr@gmail.com
ORCID: 0000-0002-3615-6926

***Abstract*— Determination of radar traces are world-widely important issue for the defense industry and civil aviation. Due to many objects such as drones, birds, higher energy plants components, there have been many artifacts to determine real flying object (i.e. target) from these surrounding factors mentioned. Therefore, a series of target classes are of critical importance in surveillance radars, which are used to manage air traffic or detect targets within the territory. In the present work, we report a comprehensive comparison of various machine learning algorithms by considering the flight data series of the objects. Different machine learning algorithms are applied to the problem by considering the statistical features the obtained 3 typical flight data, namely, real and synthetically created S band Radar Cross Section (RCS), Velocity and Altitude values. Following the analyses, it has been proven that the proposed model can diagnose the classes of air targets with good accuracies namely from 85.71% to 90.16% during the testing phase of 5 different machine learning algorithms.**

Keywords— Aerial Target Classification, Machine Learning, S Band, Radar Cross Section, Air Surveillance Radar.

I. Introduction

Design of an effective air defense system or ensuring a high level of flight safety in air traffic management either civil or defense basis is a hot and important topic. Analyzing and classifying a wide variety of object types and their different scale groupings for the air targets will contribute to better management in the radar coverage area. Algorithms developed to classify the target by processing different flight data obtained by an air surveillance radar are available in the literature. Of course, the success of the algorithms would be determined by the characteristics of the data to be used for classification in this manner.

There exist different studies in the literature in terms of parameters and methods used for the target classification. When selecting features that can discriminate in solving classification problems, the suitability of these features is taken into consideration. Kinematic features are generally among the first preferred features. In the present work, different classification algorithms based on deep learning have been discussed by using kinematic features such as velocity, acceleration, turn rate, doppler, as well as RF characteristics features such as Signal to Noise Ratio (SNR).

It has been shown that UAV, non-UAV and micro-UAV classifications can be made with these algorithms by using real test data taken from the field [1]. In a different study, it was explained that after detecting birds and drones in low altitude airspace by using motion characteristics such as average speed, standard deviation of speed and maneuverability factors, a classification can be made using the random forest model, which is one of the machine learning languages [2]. Apart from the kinematic features, one of the most commonly used features is radar cross-section (RCS). In order to classify airborne objects with sufficient accuracy, the success rates of alternative machine learning algorithms were investigated using the RCS, which is an important radar signature and is considered to be a source other than the visual sources used as a data set in different studies. The authors found that different drones can be classified with a good accuracy according to their methodology [3]. There are many studies on drones in the literature: The flying drones could be distinguished from other aerial objects as well as among themselves by analyzing the echoes reflected from a flying drone with radar and using features such as RCS as discriminators [4,5]. The distinguishability of two fighter aircrafts has been demonstrated by performing target classification with artificial neural networks over the RCS database [6]. In a recent work, Kaya and Kaplan [7] performed another study on neural networks algorithms in the light of radar data collected from drone, ship and bird targets. Their work was completed successfully with high accuracies. In a different work, Chen and the colleagues analyzes different machine learning algorithms and a data-based method was used in their scheme. It was stated that faults existing in the distribution system equipment can be identified by analyzing partial discharge defects [8].

The aim of our study presented here is to investigate algorithms that will enable the classification of objects that may be in the coverage area using an S-Band operating radar. The targets in the radar coverage area are divided into 7 different classes and defined as birds, helicopters, passenger planes, drones, UAVs, fighter aircraft and wind turbines. Initially an information on how the data is prepared for machine learning algorithms is explained in Section 2. In the next section, classification approaches are explained. The third section includes the findings with a discussion on the accuracy rates for various algorithms. Finally, the results and planned future studies are presented.

II. Data Collection and Labeling

In the present work, the class of the target was decided by using the information obtained from the S band radar for the classification problem. The data processed by the radar are the RCS, velocity and altitude values of the target as a set of information. The target identified with the number “1” is the bird, and the other numbers from “2” to “7” represent

EK-5. (devam) Makale yayını

helicopter, passenger plane, drone, UAV, fighter aircraft and wind turbine in a wide object sequence, respectively.

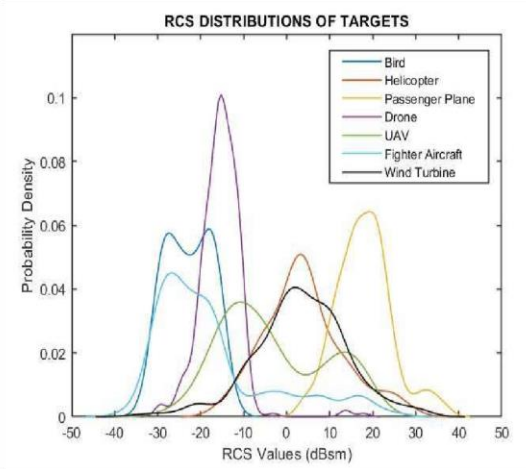


Fig. 1. The collected Radar Cross Section (RCS) data distributions.

Open information sources (articles, device catalogs, etc.) have been used to collect the data for this project. The RCS values of different targets are presented in Fig. 1. The data on velocity has been presented in Fig. 2 and finally the altitude values are shown in Fig. 3. Note that all those are presented in the corresponding plots by using kernel density estimation (ksdensity) in this context. According to all these probability densities, many different patterns are obtained. For instance, drones indicate larger density around RCS of -15, whereas passenger planes show an RCS of +18 and between them, helicopters and wind turbine blades and other UAVs exist in terms of RCS values. On the one hand, fighter aircrafts and birds have larger densities around -20 RCS. From the point of velocity data (in Fig. 2), the highest probability exists for the drones with 5 m/s average. Whereas, passenger planes and fighter aircrafts have the highest velocities.

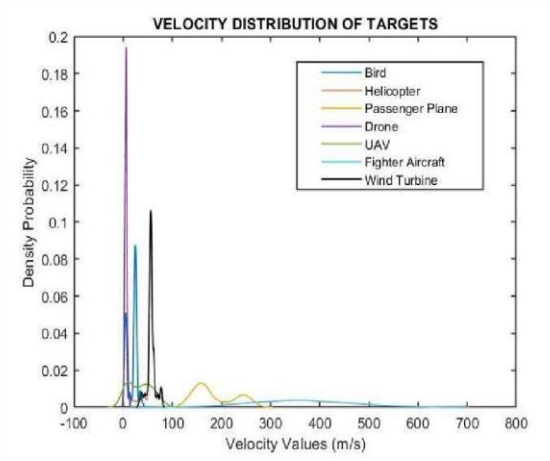


Fig. 2. The collected Velocity data distributions.

The wind turbine blades have mean velocity value of 57 m/s and helicopters and UAVs have slightly lower velocity compared to the turbine blades in this respect. According to Fig. 3, the highest probability density is obtained for drones. The other altitude values increase substantially for other target groups, i.e., helicopters, UAVs, passenger planes, fighter aircrafts. However all those have a variable altitude values starting from 100 feet to 45.000 feet. In addition, these have lower probability densities, which are differed from the drones.

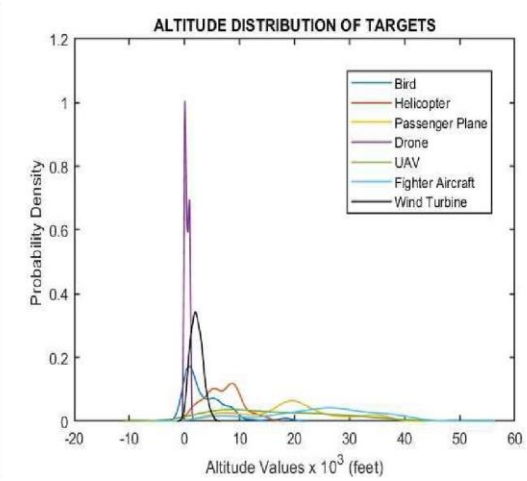


Fig. 3. The collected Altitude data distributions.

Following the visualization of the data collected with the kernel density estimation method in Figs. 1-3, we also introduce Table I, where the mean and standard deviation values of the data collected for the targets are given. In this way, the central points of the data set and how far the data set values are from the average are shown, numerically.

TABLE I. MEAN AND STANDARD DEVIATION VALUES OF THE DATA

Class Labels	Mean RCS (S Band-dBsm)	Std RCS (S Band-dBsm)	Mean Velocity (m/s)	Std Velocity (m/s)	Mean Altitude x10^3 (feet)	Std Altitude x10^3 (feet)
1	-22.79	5.17	18.33	10.33	3.77	4.01
2	3.88	9.29	55.34	11.13	6.94	3.25
3	18.02	6.36	186.18	42.29	19.09	8.74
4	-15.38	5.03	5.61	2.2019	0.44	0.37
5	-3.16	12.15	34.28	23.79	16.40	10.17
6	-17.34	13.83	361.52	96.63	24.16	9.99
7	3.26	10.72	56.84	8.43	2.19	1.02

III. CLASSIFICATION APPROACHES

A. Methodology for Machine Learning Algorithms

This subsection provides details about the steps followed in the machine learning procedure. Initially, it should be pointed out that machine learning consists of algorithms accustomed to learning new information from the available data and to create a more suitable model with this learned information. In this

EK-5. (devam) Makale yayını

manner, machine learning process uses some techniques such as supervised learning, unsupervised learning, reinforcement machine learning, etc. The underlying reason for choosing the supervised learning technique used in this study is that the output (labeled) data we are trying to predict is known in advance. A supervised learning algorithm takes a known set of input data and known responses to the data (output) and trains a model to generate reasonable predictions for the response to the new/test data. In this technique, a workflow diagram is shown in Fig. 4, which trains a model on known input and output data to predict future outputs.

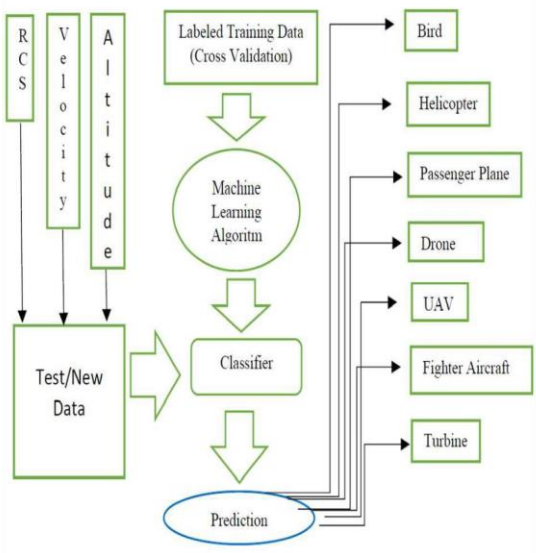


Fig. 4. The workflow diagram of supervised machine learning process.

The data sets for the supervised machine learning are generally divided into two groups: Training and test sets. Here, the training data set refers to the samples that were used to construct the model, whereas the testing data set is used to evaluate the model's performance. In this study, 300 data have been prepared from 3 flight parameter data (RCS, Velocity, Altitude) for each of 7 different classifications. The data is split into 85% training and 15% testing. Of the flight data for a total of 300 sets, 255 samples have been selected for the training set and 45 samples for the test set. Machine learning algorithms have been used for the training set containing 255 data for each flight data, and also through 5-fold cross-validation. The original data have been divided into five subsets, each using one subset as the validation set and the other four subsets as the training set. All the trained models have been obtained by repeating all these processes five times. Following the completion of the verification procedures, a test set containing 45 samples has been applied to the five models with the highest accuracy rates. The model's performance on the test set with known values can then be evaluated. The success rates of different classification algorithms are examined by using the "Classification Learner" included in Matlab applications. Common algorithms used for classification purposes include vector machines (SVMs), boosted and bagged decision trees, k-nearest neighbors, Naive Bayes, discriminant analysis, logistic regression, and neural networks in this context.

B. Accuracy and Success of Classification Algorithm Models

TABLE II. ACCURACY OF 5 MACHINE LEARNING MODELS

Algorithm Model Type	Accuracy (Validation)	Accuracy (Test)
Fine Tree	87.34 %	85.71 %
Trilayered Neural Network	87.17 %	90.16 %
SVM Kernel	88.80 %	86.98 %
Cubic KNN	84.93 %	85.71 %
Gaussian Naive Bayes	86.50 %	86.98 %

In Matlab applications, there are various machine learning algorithms, which are commonly used for the classification purposes. Depending on the data sets used, these algorithm models may have advantages over each other in terms of success, speed and ease of method in different application areas. In this study, the accuracy rates of 5 different algorithm models in the validation have been obtained during the training phase and testing phases are presented in Table II, clearly. As can be seen from the table, it would be appropriate to use Trilayered Neural Network and SVM Kernel models to have better results. The classification in machine learning is the process of separating a given set of data into different categories. In the machine learning terminology, the confusion matrix is used to measure the performance of the classification model. The ratio between correctly and incorrectly predicted samples is determined with the confusion matrix obtained from the test set, which the classification model has not been seen before and where the real label values are known. The success rates obtained using Trilayered Neural Network and SVM Kernel classification models are presented in Figs. 5 and 6, respectively.

Fig. 5. Test confusion matrix (Model Trilayered Neural Network).

EK-5. (devam) Makale yayını

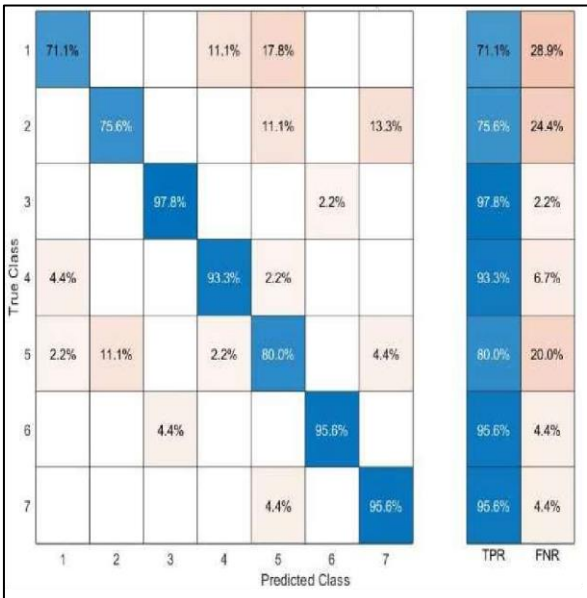


Fig. 6. Test confusion matrix (Model SVM Kernel).

In these diagrams, each column corresponds to the predicted class label, and each row corresponds to the actual class label. In this way, how many correct/incorrect classifications of the test samples are made is obvious. For example, with the Confusion Matrix of Model Trilayered Neural Network, which allows visualizing and summarizing the performance of the model, it is seen that 36 out of 45 test data belonging to class of bird which we indicate with the label number "1". It gives us that this is perfectly predicted as achieving a success rate of 80%. A success level of 90% is achieved for all the classes. When the Confusion Matrix of SVM Kernel examines the data, the success rate for the same class, is found to be 71.1% by correctly classifying 32 out of 45 targets. When we compare two models in general, it is obvious that the SVM Kernel model is successful in the passenger plane classified with label number "3", but the Trilayered Neural Network model is superior in other classes, too. Therefore, this method can be considered as a performance measurement in machine learning classifications so that it helps evaluate the accuracy and effectiveness of the model's predictions.

Receiver Operating Characteristic (ROC) analysis results, which are graphical representations showing how well the classification models perform, are presented in Figs. 7 and 8. The ROC curve is an important tool for evaluating the performance of a machine learning model. Considering that the Area Under the Curve (AUC) value is close to one, it means that the model would perform well, it seems that the models are at a level that meets the need. In the model of Trilayered Neural Network, the best value is obtained in classes with "6" labels, and in the model of SVM Kernel, in classes with "3" labels.

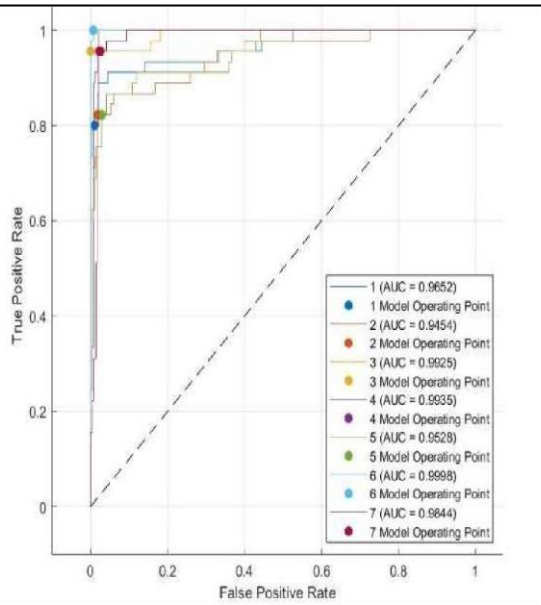


Fig. 7. ROC Curve (Model Trilayered Neural Network)

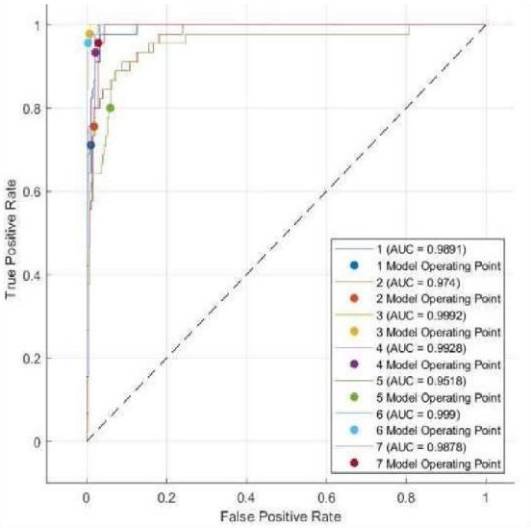


Fig. 8. ROC Curve (Model SVM Kernel)

C. *Methods of Proposed Classification Algorithms*

In this subsection, the definition of the logic underlying two different classifiers will be given. The first of these classifiers, Support Vector Machine (SVM) is part of a group of kernel-based methods which are used for pattern classification. SVM is basically a linear classifier that classifies linearly separable data, but in general, the feature vectors might not be linearly separable. To overcome this issue, kernel trick is used [9]. SVM can work with non-linearly separable data by turning originally low-dimensional data sets into high-dimensional data sets with different kernel functions such as linear, polynomial, radial basis function, sigmoid. Thus, if datasets can be converted to high dimensions, it is possible to classify them as linear. In Fig. 9, the conversion of linearly non-separable data into linearly separable data with Kernel functions is visualized [10].

EK-5. (devam) Makale yayını

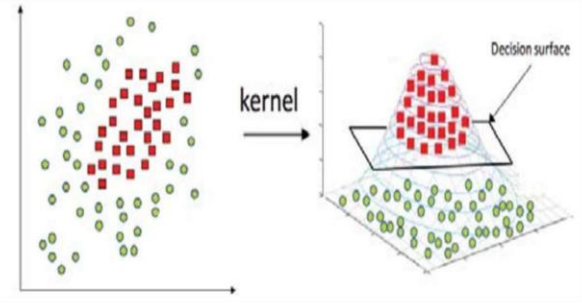


Fig. 9. Kernel Mapping Function

The optimum hyperplane (decision boundaries) is determined with the SVM algorithm applied after the dimension change, and the hyperplane forms the decision mechanism between the two classes.

The second classifier, a neural network is used for different purposes in many different fields. One of the areas, where a neural network is used is to process data as a machine learning model and classify them under different categories. It makes predictions on unlabeled data using the information learned from labeled training data. The main purpose of a neural network model is to try to find the relationship between features in a data set. It mimic human brain processing. Each connection between neurons as a mathematical function that improves the network's ability to make accurate predictions and classifies information (data) according to special algorithms. The output of each neuron is determined by using an activation function such as sigmoid, (Rectified Linear Unit) ReLU [9].

The neural network consists of interconnected nodes, or neurons, organized in layers. The neurons interconnection link carries certain weight. The input layer receives the initial data, which is then processed through hidden layers before producing an output in the final layer. In this study, the neural network algorithm we specifically examined has 3 layers, each size 10. In our model used, ReLU activation function for layers was preferred. The three-layer neural network structure is presented in Fig. 10 [11].

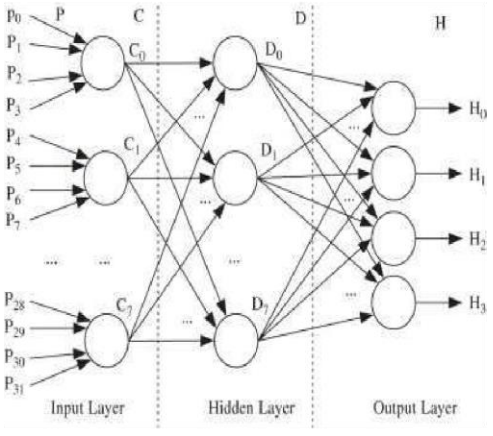


Fig. 10. The three-layer Neural Network Structure.

The direct comparison between other classifiers may not be possible since neural networks are nonlinear model-free method while statistical methods are basically linear and model based [12-15].

IV. Conclusion

In this study, machine learning algorithms solved the target classification problem. In particular, the high performance of Trilayered Neural Network and the SVM Kernel models in Matlab Applications was observed. Accuracy rates of 90.16 and 86.98 were obtained from these two machine learning algorithms, respectively. The variables that feed the algorithms were selected as RCS, velocity and altitude. The distribution of these variables in different classes (target types) constituted the data set.

As a result of testing the proposed models, they have been confirmed that they make consistent, fast and highly accurate decisions. This is visualized with the Confusion matrix (Fig-5 and 6), which is graph that summarizes the performance of a classification model/algorithm for machine learning processes. Thus, it has been seen that machine learning-based classifications can be used in air target classification applications. Considering the obtained results, air traffic controllers who are air surveillance radar users will be able to manage air traffic more reliably and ensure airspace security. In the future, it is planned to collect more data, increase the number of targets to be classified (7 in this study), and improve the accuracy rate by increasing the efficiency in the training phase.

References

[1] T.B. Sarikaya, D. Yumus, M. Efe, G. Soysal, and T. Kirubarajan, "Track based UAV classification using surveillance radars," in *Proc. 22th Int. Conf. Inf. Fusion*, Jul. 2019, pp. 1–6.

[2] J. Liu, Q.Y. Xu, and W.S. Chen, "Classification of birds and drone targets based on motion characteristics and random forest model using surveillance radar data," *IEEE Access*, vol. 9, pp. 160135- 160144, 2021.

[3] S. Roychowdhury and D. Ghosh, "Machine learning based classification of radar signatures of drones," in *Proc. IEEE 2nd Int. Conf. Range Technol.*, 2021, pp. 1–5.

[4] B.R. Phanindra, R.N. Pralhad, and A.A.B. Raj, "Machine learning based classification of ducted and non-ducted propeller type quadcopter," In 2020 *6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pp. 1296-1301. IEEE, 2020.

[5] R. Nakamura and H. Hadama, "Characteristics of ultra-wideband radar echoes from a drone," *IEICE Commun. Express*, vol. 6, no. 9, pp. 530–534, Sept. 2017.

[6] E. Aydemir, and M. Gose, "Radar cross section identification of air targets using the Cosine transform and neural networks," *Recent Patents on Engineering*, vol. 6.1, pp. 65-69, 2012.

[7] E. Kaya and G.B. Kaplan, "Neural network based drone recognition techniques with non-coherent S-band radar", *Proc. IEEE Radar Conf. (RadarConf)*, pp. 1-6, May 2021.

[8] J. Chen, Y. Li, J. Chen, D. Liu, Y. Yang, "Identification of typical partial discharge defects of distribution system equipment based on classification learner," In *Proc. the 2022 7th Asia Conference on Power and Electrical Engineering (ACPEE)*, Hangzhou, China, 15–17 April 2022; pp. 2025–2029.

[9] M. Hussain, S. Kanwal Wajid, A. Elzaart, and M. Berbar, "A comparison of svm kernel functions for breast cancer detection," 2011 *Proc. 8.*

EK-5. (devam) Makale yayını

International Conference Computer Graphics, Imaging and Visualization. 145 – 150 (2011).

[10] O. Karal, "Performance comparison of different kernel functions in SVM for different k value in k-fold cross-validation," in: *Proc. Innov. Intell. Syst. Appl. Conf., IEEE*, 2020: pp. 1–5.

[11] S. Lian, J. Sun, Z. Wang "Secure hash function based on neural network" *Neurocomputing*, 69 (2006), pp. 2346-2350.

[12] G. Zhang, "Neural networks for classification: A survey," *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 4, pp. 451–462, Nov. 2000.

[13] E. Çelik, H. Gör, N. Öztürk, and E. Kurt, "Application of artificial neural network to estimate power generation and efficiency of a new axial flux permanent magnet synchronous generator," *Int. J. Hydrogen Energy*, 42(28), (2017), 17692–17699, doi:10.1016/j.ijhydene.2017.01.168

[14] B. Dursun, Y. Uzun, and E. Kurt, "The efficiency estimation of 900 MHz RF energy harvester using artificial neural network," Electrical and Electronics Engineering, 06(03), 17–23, (2017) doi:10.14810/elelij.2017.6303

[15] H. Ateş, B. Dursun, and E. Kurt, Estimation of mechanical properties of welded S355J2+N steel via the artificial neural network," Scientia Iranica, 23(2), 609–617, (2016) doi:10.24200/sci.2016.3848

Gazili olmak ayrıcalıktır