T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

HAVA LİDAR VERİLERİN YAPAY ZEKA YÖNTEMLERİ KULLANILARAK SINIFLANDIRILMASI VE FİLTRELENMESİ

Burcu BAYASLI

YÜKSEK LİSANS TEZİ

Harita Mühendisliği Anabilim Dalı

Uzaktan Algılama ve CBS Programı

Danışman

Dr. Öğr. Üyesi Alper ŞEN

Mayıs, 2021

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

HAVA LİDAR VERİLERİNİN YAPAY ZEKA YÖNTEMLERİ KULLANILARAK SINIFLANDIRILMASI VE FİLTRELENMESİ

Burcu BAYASLI tarafından hazırlanan tez çalışması 29.05.2021 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Harita Mühendisliği Anabilim Dalı, Uzaktan Algılama ve CBS Programı **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Dr. Öğr. Üyesi Alper ŞEN
Yıldız Teknik Üniversitesi
Danışman

Jüri Üyeleri

Dr. Öğr. Üyesi Alper ŞEN, Danışman
Yıldız Teknik Üniversitesi _____

Prof. Dr. Ali Melih BAŞARANER, Üye
Yıldız Teknik Üniversitesi _____

Prof. Dr. Tarık TÜRK, Üye
Sivas Cumhuriyet Üniversitesi _____

Danışmanım Dr. Öğr. Üyesi Alper ŞEN sorumluluğunda tarafımca hazırlanan “Hava Lidar Verilerinin Yapay Zeka Yöntemleri Kullanılarak Sınıflandırılması ve Filtrelenmesi” başlıklı çalışmada veri toplama ve veri kullanımında gerekli yasal izinleri aldığımı, diğer kaynaklardan aldığım bilgileri ana metin ve referanslarda eksiksiz gösterdiğimi, araştırma verilerine ve sonuçlarına ilişkin çarpıtma ve/veya sahtecilik yapmadığımı, çalışmam süresince bilimsel araştırma ve etik ilkelerine uygun davrandığımı beyan ederim. Beyanımın aksinin ispatı halinde her türlü yasal sonucu kabul ederim.

Burcu BAYASLI

İmza

Sevgili Aileme,

TEŞEKKÜR

Lisans ve yüksek lisans sürecim boyunca tez çalışmalarıma büyük bir özveriyle destek veren ve bana yol gösteren danışmanım Dr. Öğr. Üyesi Alper Şen'e, Almanya'da Erasmus sürecim boyunca bana destek olan Leibniz Hannover Üniversitesi'ndeki danışmanım Prof. Dr. -Ing. habil. Monika Sester'a, hem tezimi hem de kendimi geliştirmem için önemli katkılar veren MSc. Florian Politz'e ve Bergama hava Lidar verisini bizimle paylaşan Harita Genel Müdürlüğü'ne saygılarımı sunar ve teşekkürü bir borç bilirim.

Yüksek lisansı ve işimi bir arada ilerletmemi destekleyen Orkut Murat Yılmaz'a, anlayışları için Yer Çizenler'deki ekip arkadaşlarıma, işini gücünü bırakıp her sorunumda beni dinleyen Tümay Arda'ya ve en yorucu zamanlarda başımı omzuna yasladığım İsa Tuzer'e tüm samimiyetimle teşekkür ederim.

Beni bugünlere getiren, her koşulda tüm kararlarımın arkasında duran, sevgisini ve ilgisini hiçbir zaman eksik etmeyen, maddi-manevi en büyük desteği sağlayan aileme sonsuz sevgilerimi sunarım.

Burcu BAYASLI

İÇİNDEKİLER

SİMGE LİSTESİ vii

KISALTMA LİSTESİ ix

ŞEKİL LİSTESİ ix

TABLO LİSTESİ xii

ÖZET xiii

ABSTRACT xv

1 GİRİŞ **1**

1.1 Literatür Özeti 1

1.2 Tezin Amacı 6

1.3 Hipotez 6

2 HAVA LİDAR VERİSİ **8**

2.1 Hava Lidar Verilerinin Elde Edilişi 8

2.2 Hava Lidar Veri Yapısı 9

3 VERİ BİLİMİNE GİRİŞ VE YAPAY ZEKA **11**

3.1 Makine Öğrenmesine Giriş 12

3.2 Rastgele Orman Algoritması (Random Forest Algorithm) 13

3.3 Yapay Sinir Ağlarına Giriş 16

3.4 Kohonen'in Kendini Düzenleyen Haritaları 21

3.5 Derin Öğrenmeye Giriş 24

4 ÇALIŞMA ALANI VE VERİSETİ ÖZELLİKLERİ **38**

5 METODOLOJİ **42**

5.1 Kendini Düzenleyen Haritalar 42

5.2 Rastgele Orman Algoritması 44

5.3 Derin Öğrenme 46

6 KULLANILAN ARAÇLAR VE PARAMETRELER **52**

6.1 Kendini Düzenleyen Haritaların Uygulanması 53

6.2 Rastgele Orman Algoritmasının Uygulanması 54

6.3 Derin Öğrenmenin Uygulanması 55
7 BULGULAR 57
7.1 Kendini Düzenleyen Haritalar 57
7.2 Rastgele Orman Algoritması 65
7.3 Derin Öğrenme 73
8 SONUÇ VE ÖNERİLER 81
KAYNAKÇA 83
A ÜÇ YÖNTEMİN KARŞILAŞTIRILDIĞI TABLO 89
TEZDEN ÜRETİLMİŞ YAYINLAR 90

SİMGE LİSTESİ

w	Ağırlık parametresi
b	Eşik değer
m	Fonksiyonun eğimi
$h_{ci}(t)$	Gauss modeli
x_K	Girdi vektörü
m_i	Her nörona ait referans vektörü
$\sigma(t)$	Kernel yarıçapı
h_{ci}	Komşuluk fonksiyonu
γ	Komşuluk yarıçapı
X_{max}	Maksimum X değeri
m_c	Minimum mesafe referans vektörü
X_{min}	Minimum X değeri
ρ	Ortalama nokta yoğunluğu
ε_i	Öznitelik değeri
μ_i	Rastgele öznitelik değeri
$\mathbb{R}$	Reel sayılar
$\sigma(z)_x$	Softmax fonksiyonu
d_{ci}	Topolojik mesafe
k	Yığın boyutu
$\Delta x, y$	Yığın örnekleme mesafesi
t	Zaman

KISALTMA LİSTESİ

ALS	Airborne Lidar System (Hava Lidar Sistemi)
CNN	Convolutional Neural Networks (Evrişimli Sinir Ağları)
DIN	Dense Image Matching (Yoğun Görüntü Eşleştirme)
DN	Doğru negatif
DP	Doğru pozitif
GPS	Global Positioning System (Küresel Konumlama Sistemi)
IMU	Inertial Measurement Unit (İnersiyal Ölçme Ünitesi)
ISPRS	International Society for Photogrammetry and Remote Sensing (Uluslararası Fotogrametri ve Uzaktan Algılama Derneği)
KDH	Kendini Düzenleyen Haritalar
LİDAR	Light Detection and Ranging (Lazer ile Yüzey Uzaklıklarının Ölçümü)
LSTM	Long Short Term Memory (Uzun Kısa Süreli Bellek)
RGB	Red-Green-Blue (Kırmızı-Yeşil-Mavi)
RO	Rastgele Orman
RPN	Region Proposal Network (Bölge Öneri Ağı)
SYM	Sayısal Yükseklik Modeli
TIN	Triangulated Irregular Network (Üçgenlenmiş Düzensiz Ağ)
YN	Yanlış negatif
YP	Yanlış pozitif
YSA	Yapay Sinir Ağları
2B	İki Boyutlu
3B	Üç Boyutlu

ŞEKİL LİSTESİ

Şekil 2.1 Hava Lidar sisteminin çalışma prensibi 8
Şekil 2.2 Hava Lidarın çoklu yansıma özelliği 9
Şekil 2.3 Ayrık dönüşlü Lidar ve tam dalga formlu Lidar 10
Şekil 3.1 Veri bilimi bileşenleri 11
Şekil 3.2 Sık kullanılan yapay zeka yöntemlerinin birbirleriyle ilişkisi 12
Şekil 3.3 Makine öğrenmesinin alt dalları içerisinde derin öğrenme, kendini düzenleyen haritalar ve rastgele orman algoritmasının yeri 13
Şekil 3.4 Bir karar ağacının yapısı 14
Şekil 3.5 Rastgele orman algoritmasının çalışma şekli 15
Şekil 3.6 Tek katmanlı algılayıcı 16
Şekil 3.7 Çok katmanlı ileri beslemeli geri yayılımlı yapay sinir ağları 17
Şekil 3.8 Aktivasyon fonksiyonları 19
Şekil 3.9 Sigmoid ve Softmax aktivasyon fonksiyonlarının kıyaslanması 19
Şekil 3.10 Kayıp fonksiyonu, global ve yerel minimum, gradyan 21
Şekil 3.11 Kazanan vektörün girdi vektörüne göre değişimi ve eğitim boyunca azalan doğruluk yarı çapı 22
Şekil 3.12 Altıgen topolojiye sahip x-y yönlerinde nöron ağı 23
Şekil 3.13 KDH Eğitimi 24
Şekil 3.14 Geleneksek makine öğrenmesi ve derin öğrenme arasındaki akış diyagramı farkı 24
Şekil 3.15 Derin öğrenmenin yaygın kullanılan bileşenleri 24
Şekil 3.16 Seyreltme 26
Şekil 3.17 Filtre ve görüntü: Sağda 3x3'lük, derinliği 1, kademesi 1 ve dolgulaması olmayan filtre, solda 5x5'lik görüntü 27
Şekil 3.18 CNN filtreleme işlemi 27
Ş**ekil 3.19** Dolgulama ile CNN filtrelemesi 28
Şekil 3.20 Havuzlama yöntemleri 29
Şekil 3.21 Nokta bulutu sınıflandırmada sıkça kullanılan CNN yöntemleri 29
Şekil 3.22 CNN Özellikleri ile Bölge Çıkarımı ve Bölgelerin Sınıflandırılması 31
Şekil 3.23 VoxNet 32
Şekil 3.24 PointNet'in uygulama alanları 34

Şekil 3.25 ALSNet çalışma şekli 36
Şekil 4.1 Birinci çalışma alanının 2013 yılına ait uydu görüntüsü 38
Şekil 4.2 Birinci çalışma alanının 2015 yılına ait sokak görüntüsü (tepenin yamacındaki binalar) 39
Şekil 4.3 Birinci çalışma alanına ait Lidar verisi 38
Şekil 4.4 İkinci çalışma alanının 2013 yılına ait uydu görüntüsü 39
Şekil 4.5 İkinci çalışma alanı içerisinden bir sokağın 2015 yılına ait sokak görüntüsü 39
Şekil 4.6 İkinci çalışma alanına ait Lidar verisi 41
Şekil 5.1 KDH iş akış diyagramı 44
Şekil 5.2 RO algoritmasının iş akış diyagramı 45
Şekil 5.3 k-Boyutlu ağaçların çalışma şekli 46
Şekil 5.4 Dönüklük matrisi 48
Şekil 5.5 Derin öğrenmede kullanılan girdi versi ve çıktı veri 48
Şekil 5.6 Derin öğrenme iş akış diyagramı 49
Şekil 5.7 Hata matrisi örneği 51
Şekil 6.1 Kullanılan yazılımlar ve programlama dilleri 52
Şekil 6.2 KDH topolojisi 53
Şekil 6.3 Eğitim ve test veri setlerinin ayrılması 54
Şekil 7.1 Birinci çalışma alanına ilk uygulanan KDH sonrası sonuçlar 57
Şekil 7.2 Birinci çalışma alanına ikinci kez KDH uygulandıktan sonra yakalanan bazı binalar 58
Şekil 7.3 Birinci çalışma alanına ikinci kez uygulanan KDH sonrası sonuçlar 59
Şekil 7.4 KDH yöntemiyle sınıflandırılan zemin, bina ve bitki örtüsü noktaları 61
Şekil 7.5 Birinci alanın KDH ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi 62
Şekil 7.6 İkinci çalışma alanının KDH ile son sınıflandırma sonuçları 64
Şekil 7.7 İkinci çalışma alanının KDH ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi 65
Şekil 7.8 RO algoritması ile sınıflandırma sonuçları: (A) manuel, (B) RO 66
Şekil 7.9 RO algoritması ile sınıflandırma sonuçları (perspektif) (A)manuel (B)RO 67
Şekil 7.10 Birinci alanın RO algoritması ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi 68

Şekil 7.11 Birinci çalışma alanı için x,y,z,i değerlerinin sınıflandırmayı etkileme oranları 69

Şekil 7.12 İkinci çalışma alanının RO algoritması ile sınıflandırma sonuçları (A)manuel (B)RO 70

Şekil 7.13 İkinci alanın RO algoritması ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi 73

Şekil 7.14 İkinci çalışma alanı için x,y,z,i değerlerinin sınıflandırmayı etkileme oranları 73

Şekil 7.15 Birinci çalışma alanının derin öğrenme ile sınıflandırma sonucu (A)tepeden görünüm (B)perspektif görünüm 75

Şekil 7.16 Birinci alanın derin öğrenme ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi 76

Şekil 7.17 İkinci çalışma alanının derin öğrenme ile sınıflandırma sonucu (A)tepeden görünüm (B)perspektif görünüm 78

Şekil 7.18 İkinci alanın derin öğrenme ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi 79

Şekil 8.1 Hata kaynaklarına örnek sokak görüntüleri 82

TABLO LİSTESİ

Tablo 3.1 PointNet, PointNet++ ve ALSNet için girdi verileri hazırlarken kullanılan yığın yöntemlerinin kıyaslanması 37

Tablo 7.1 Birinci çalışma alanı KDH ile sınıflandırma hata tablosu 60

Tablo 7.2 Birinci çalışma alanı KDH ile filtreleme hata tablosu 60

Tablo 7.3 İkinci çalışma alanı KDH ile sınıflandırma hata tablosu 63

Tablo 7.4 İkinci çalışma Alanı KDH ile filtreleme hata tablosu 63

Tablo 7.5 Birinci çalışma alanı RO algoritması ile sınıflandırma hata tablosu 67

Tablo 7.6 Birinci çalışma alanı RO algoritması ile filtreleme hata tablosu 68

Tablo 7.7 Birinci çalışma alanı RO algoritması ile sınıflandırma doğruluk ölçütleri 69

Tablo 7.8 İkinci çalışma alanı RO algoritması ile sınıflandırma hata tablosu 71

Tablo 7.9 İkinci çalışma alanı RO algoritması ile filtreleme hata tablosu 71

Tablo 7.10 İkinci çalışma alanı RO algoritması ile sınıflandırma doğruluk ölçütleri 73

Tablo 7.11 Birinci çalışma alanı Derin Öğrenme ile sınıflandırma hata tablosu 75

Tablo 7.12 Birinci çalışma alanı Derin Öğrenme ile filtreleme hata tablosu 76

Tablo 7.13 Birinci çalışma alanı Derin Öğrenme ile sınıflandırma doğruluk ölçütleri 77

Tablo 7.14 İkinci çalışma alanı Derin Öğrenme ile sınıflandırma hata tablosu 78

Tablo 7.15 İkinci çalışma alanı Derin Öğrenme ile filtreleme hata tablosu 79

Tablo 7.16 İkinci çalışma alanı derin öğrenme ile sınıflandırma doğruluk ölçütleri 80

ÖZET

Hava LİDAR Verilerinin Yapay Zeka Yöntemleri Kullanılarak Sınıflandırılması ve Filtrelenmesi

Burcu BAYASLI

Harita Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Danışman: Dr. Öğr. Üyesi Alper ŞEN

Hava Lidar (Light Detection and Ranging) sistemleri ile üretilen mekânsal veriler, yüksek doğruluklu, hızlı ve az maliyetli olarak elde edilmektedir. Ancak verilerin nesne çıkarımı amacıyla elle işlenmesi, yoğun emek ve zaman gerektiren bir işlemdir. Bu süreci otomatik bir hale dönüştürmek amacıyla, denetimli/denetimsiz sınıflandırma yöntemleri kullanılabilmektedir. Lidar verilerinin, zemine ait ve zemine ait olmayan veriler olarak ayrılmasına filtreleme denir. Lidar verileri kullanılarak Sayısal Yükseklik Modeli oluşturulmasında filtreleme işlemi büyük önem arz etmektedir. Ayrık dönüşlü Lidar verisinin kullanıldığı bu çalışma, Bergama ilçesine ait iki farklı alanda yürütülmüştür. Lidar nokta bulutu, denetimsiz bir yapay sinir ağı yöntemi olan Kendini Düzenleyen Haritalar (KDH), denetimli bir makine öğrenmesi yöntemi olan Rastgele Orman Algoritması ve denetimli bir yapay sinir ağı yöntemi olan Derin Öğrenme ile hem sınıflandırılmış hem de filtrelenmiştir, ardından da yöntemler her iki çalışma alanı için de kıyaslanmıştır. Referans veri olarak tüm noktalar uydu görüntüleri ve sokak görüntüleri yardımıyla manuel olarak sınıflandırılmıştır. Bu, hem sınıflandırma doğruluğunu kontrol etmek için hem de rastgele orman algoritması ve derin öğrenmede eğitim ve test verisinin etiketleri için kullanılmıştır. KDH uygulamasında eğitim ve test olarak veri seti ayrılmadan tüm noktalar sınıf etiketsiz olarak kümelenmişlerdir. Bu nedenle KDH

uygulamasında tüm noktalar için bir sınıf ataması ve doğruluk analizi söz konusu iken, rastgele orman ve derin öğrenmede yalnızca test alanlarının sınıf ataması değerlendirilmeye alınmıştır. Genel filtreleme sonuçları en iyi derin öğrenme ile alınmıştır. Filtreleme doğrulukları birinci çalışma alanı için %91.79 ve ikinci çalışma alanı için %95.82 olarak hesaplanmıştır. Genel sınıflandırma doğruluğunda ise en iyi sonuçlar derin öğrenmeyle alınsa da nesnelerin kendi içerisindeki sınıflandırma doğruluğu yöntemden yönteme değişiklikler göstermiştir. Derin öğrenmeyle genel sınıflandırma doğrulukları birinci çalışma alanı için %87.96 ve ikinci çalışma alanı için %91.71 olarak hesaplanmıştır. KDH ve rastgele orman algoritması için her iki çalışma alanında sınıflandırma ve filtreleme doğrulukları ise %80-%90 arasında değişmektedir.

Anahtar Kelimeler: Lidar, ysa, kendini düzenleyen haritalar, rastgele orman, derin öğrenme, kdtree, denetimli öğrenme, denetimsiz öğrenme

ABSTRACT

Classification and Filtering of Airborne LİDAR Data by using Artificial Intelligence Methods

Burcu BAYASLI

Department of Geomatics Engineering

Master of Science Thesis

Advisor: Assist. Prof. Dr. Alper ŞEN

Spatial data produced by Airborne Lidar (Light Detection and Ranging) systems are obtained with high accuracy, speed and low cost. However, manual processing of data for object extraction is a labor-intensive and time-consuming process. In order to automate this process, supervised / uncontrolled classification methods can be used. Separating Lidar data into ground and non-ground data is called filtering. Filtering is of great importance in creating Digital Elevation Model using Lidar data. This study, in which discrete-turn Lidar data was used, was conducted in two different study areas of Bergama district. "Lidar point cloud was both classified and filtered using Self-Organizing Maps (SOM) -an uncontrolled artificial neural network method-, Random Forest Algorithm- a supervised machine learning method-, and Deep Learning -a supervised artificial neural network method- then these three methods have also been compared for both study fields. In order to check the classification accuracy and use the labels of training and test data in random forest and deep learning algorithms, all points were classified manually with the help of satellite and street level imageries as reference data. In the SOM application, all points were clustered without class labels and without separating the data set as training and test. That's why, class assignment of all points in the whole area were evaluated in the SOM application. However, only the class assignment of the test

areas were evaluated in random forest and deep learning algorithms. Overall filtering results are best obtained with deep learning. Overall filtering accuray was calculated %91.79 for the first study area and %95.82 for the second study area. In general classification accuracy, although the best general results are obtained with deep learning, the classification accuracy of the objects within themselves has changed from method to method. The general classification accuracies with deep learning were calculated as %87.96 for the first study area and %91.71 for the second study area. General classification and filtering accuracies vary between 80% and 90% in both study areas for SOM and random forest algorithm.

Keywords: Lidar, AI, self-organizing maps, random forest, deep learning, kdtree, supervised learning, unsupervised learning

1
GİRİŞ

1.1 Literatür Özeti

Günümüzde hava Lidar (Light Detection and Ranging) sistemleri 3 boyutlu (3B) kent modellemeleri, kıyı çizgisi tespiti, peyzaj, demiryolu güzergâhları, enerji nakil hatları belirlenmesi, çevre kirliliği modellemeleri, orman alanlarının saptanması ve hatta ağaç türlerinin belirlenmesi gibi pek çok çalışmada kullanılmaktadır. Lidar teknolojisi ile veri üretimi yüksek doğruluk, hız ve zaman açısından büyük avantaj sağlamaktadır. Buna karşın, mekânsal bilgilerin çıkarımı amacıyla Lidar verilerinin elle ve/veya otomatik olmayan yöntemlerle işlenmesi kapsamlı, maliyetli ve zaman alıcıdır. Verilerden nesne çıkarımı sürecini otomatik bir hale dönüştürmek amacıyla, denetimli/denetimsiz sınıflandırma yöntemleri kullanılabilmektedir.

Lidar verilerinin, zemine ait ve zemine ait olmayan veriler olarak ayrılmasına filtreleme denir. Lidar verileri kullanılarak, Sayısal Yükseklik Modeli (SYM) oluşturulmasında filtreleme işlemi önem arz etmektedir (Briese 2010; Kang, Liu, & Lin, 2014). Briese (2010) filtreleme algoritmalarını dört farklı kategoride gruplandırmıştır. Bunlar; yüzey tabanlı filtreleme, morfolojik filtreleme, segmentasyon tabanlı filtreleme ve aşamalı düzensiz üçgen ağı (Triangulated Irregular Network – TIN) yoğunlaştırmadır.

Filtreleme algoritmalarının kendine has avantajları ve dezavantajları vardır. Genellikle yüzey tabanlı filtreleme algoritmaları kullanılmaktadır ve çoğu arazi koşulunda tatmin edici sonuçlar vermektedir. Bununla birlikte, bu tür filtreleme algoritmaları arazi detaylarını korumakta ve bazen küçük nesneleri sınıflandırmakta zorluk çekmektedir (Mongus & Žalik, 2014). Yüzey tabanlı filtreleme algoritmasının aksine, morfoloji tabanlı filtreleme algoritmaları arazinin morfolojik ayrıntılarını korur. Segmentasyon tabanlı algoritmalar, farklı boyut ve şekillerdeki nesneleri algılamak için farklı özellikler kullanmaktadır. Bu nedenle, bu algoritmalar kentsel alanlar için uygundur. Bununla birlikte, bu algoritmalar dağınık

lazer darbeleri nedeniyle yoğun ormanlık alanlarda zorluk çekmektedir (Chen, Gao, & Devereux, 2017). Aşamalı düzensiz üçgen ağı yoğunlaştırma algoritması, ortalama doğruluk açısından genel filtreleme yöntemlerinde en iyi sonucu elde eder (Sithole & Vosselman, 2004). Bununla birlikte, bu algoritma keskin sırtlar gibi süreksiz arazileri tespit etmekte zorlanmaktadır (Chen, Wang, Zhang, Sun, & Liu, 2016).

ISPRS WG3/III çalışma grubu tarafından farklı filtreleme algoritmalarının performansları değişik topografya özelliklerine sahip test verilerinden yararlanılarak test edilmiştir. Kraus vd. (1998), Pfeifer vd. (1998, 1999a, 1999b, 2001), Kraus vd. (2001), Briese vd. (2001), Elmqvist (2001a, 2001b), Brovelli (2002) ve Wack (2002) yüzey ve segmantasyon tabanlı filtreler; Axelsson (1999, 2000, 2001) ve Sohn (2002) aşamalı TIN yoğunlaştırma tabanlı filtreler; Roggero (2001) ve Sithole (2001) morfolojik ve segmantasyon tabanlı filtreler kullanmışlardır. Örneğin, 11 numaralı test verisi bitki örtüsü ve dik yamaçlarda binalar içermektedir. Çalışmada kullanılan filtreleme yöntemlerinde çoğu nesne iyi filtrelenmiştir. Merkezdeki yol, Roggero filtresi tarafından filtrelenebilmiştir. Sohn ve Sithole filtresi tarafından çok dik eğimler filtrelenebilmiştir. Teraslar, Elmqvist, Pfeifer, Roggero ve Sithole filtresi tarafından filtrelenebilmiştir. Kullanılan filtreleme yöntemlerinde Tip-1 hatası %15.96 ile %62 arasında değişirken; Tip-2 hatası %2.4120 ile %12.17 arasında değişmiştir. Toplam hata ise %10.76 ile %36.96 arasında değişmiştir (Sithole & Vosselman, 2003).

Biyolojik sinir ağlarından esinlenerek makine ve bilgisayarlara öğrenebilme, tahmin edebilme ve karar verebilme yetileri sağlayan yapay zekâ ise günümüzde pek çok karmaşık problemi çözebilmektedir. Denetimsiz bir sınıflandırma yöntemi olan Kendini Düzenleyen Haritalar (KDH), denetimli sınıflandırma yöntemlerinden çok katmanlı yapay sinir ağları, Rastgele Orman algoritması ve Derin Öğrenme gibi yapay zeka yöntemlerinin kullanıldığı Lidar sınıflandırma ve filtreleme çalışmaları aşağıda verilmiştir.

Morris vd. (2005), Lidar yükseklik verisini, multispektral görüntüyü ve yapay sinir ağlarını entegre ederek Güney Carolina'da bataklık karakterizasyonunu araştırmıştır.

Salah vd. (2009), multispektral hava görüntüleri ve Lidar verisinin entegrasyonundan KDH yöntemiyle otomatik nesne çıkarmış ve karakteristiği farklı iki bölgede test etmiştir. Elde edilen sonuçlar, KDH yöntemiyle Lidar verisinin, yalnızca hava görüntülerinin kullanılmasına kıyasla nesne algılama doğruluğunu %38 oranında geliştirdiğini göstermiştir.

Zaletnyik vd. (2010), KDH yöntemini kullanarak Lidar verisinin dalga formunu sınıflandırmış ve bunun için geri yansıyan dalga formunun parametrelerine dayanarak kütle merkezinden hesaplanan uzaklıklar ile bitki örtüsüz yüzeyi kaldırım ve çatı kategorilerine ayırmak için kullanmışlardır.

Grebby vd. (2011) yine hava Lidar ve multispektral görüntüyü entegre etmiş ve bitki örtüsü olan alandaki litolojik sınıflandırma için KDH yöntemini kullanmıştır.

Giampouras vd. (2013), yapay sinir ağları metoduna başvurarak hiperspektral özellikle entegre edilmiş Lidar verisi ile arazi örtüsünün sınıflandırılmasını gerçekleştirmişlerdir.

Kwon vd. (2017), ortofoto ve Lidar verilerinden çok katmanlı yapay sinir ağlarını kullanarak orman alanlarını çıkartmış ve saha ölçümleri ile karşılaştırıldığında %70 oranında bir başarı sağlamıştır.

Şen vd. (2020), iki farklı bölge için KDH yöntemini kullanarak ayrık-dönüşlü hava Lidar verilerini 3B koordinat ve yansıma özniteliğini kullanarak sınıflandırmıştır.

Denetimli öğrenme yöntemlerinden Rastgele Orman algoritmasını kullanarak Chehata vd. (2009), tam dalga formu Lidar verisiyle kentsel alanda öznitelik seçimi ve nesne çıkarımı gerçekleştirmişlerdir. Çalışmanın amacı, kentsel alanları sınıflandırmak için en uygun özellikleri seçmek ve az sayıda özellikle doğru bir sınıflandırma sağlamak olarak öne sürülmüştür. Seçilen özniteliklerle kentsel alandaki noktalar bina, bitki örtüsü, yapay zemin ve doğal zemin olmak üzere dört etikette sınıflandırılmıştır. Genel doğruluk %94.35 olarak hesaplanmıştır.

Niemeyer vd. (2012), benzer şekilde kentsel alana ait nesneleri sınıflandırmak için Rastgele Orman algoritmasıyla belirli özellikleri seçip, koşullu rastgele alanlar oluşturmuştur. Bütünlük ve doğruluk ölçütleri baz alınarak üç farklı çalışma

alanında bina sınıflandırma kalitesi %79.4, 86.2 ve %88.3, ağaçların sınıflandırılma kalitesi %35.0, %58.2 %47.9 olarak hesaplanmıştır.

Huang & Zhu (2013), bitki örtüsü sınıflandırmasında Lidar verilerini ve hiperspektral görüntülerini birleştirmek için Rastgele Orman algoritmasına dayalı bir entegrasyon yöntemi önermişlerdir. Rastgele Orman algoritmasının sınıflandırma yaparken hangi özelliklerin daha önemli olduğunu tahmin edebilme yetisinden faydalanılarak iki farklı veri kaynağından gelen spektrum, yükseklik ve bunlara karşılık gelen dokulardan gelen bilgilerin entegrasyonu sağlanmıştır.

Chen vd. (2014), Çin'in Three Gorges bölgesinde Lidar verilerini ve Rastgele Orman algoritmasını kullanarak ormanlık alandaki heyelan tespiti üzerine yaptığı çalışmada, test alanı için %78.24'lük bir ortalama genel sınıflandırma doğruluğu elde etmiştir.

Ma vd. (2017), Çin'in Miyun bölgesinde tam dalga boylu Lidar verisini Rastgele Orman algoritması ile binalar, ağaçlar, tarlalar ve zemin olmak üzere sınıflandırarak %89.73 oranında genel doğruluk elde etmiştir.

Sevgen (2018), Bergama'nın karmaşık bir kentsel alanına ait hava Lidar verisini Rastgele Orman algoritmasıyla bina, bitki örtüsü, asfalt yol ve zemin noktaları olmak üzere sınıflandırmış ve sırasıyla %77.90, %58.37, %72.90, ve %71.53 sınıflandırma doğruluklarını elde etmiştir. Sınıflandırmanın genel doğruluğunu ise %70.20 olarak hesaplamıştır.

Wang vd. (2019), kentsel Lidar verileri için piksel karşılaştırma özelliklerine dayanan bir Rastgele Orman sınıflandırma algoritması geliştirmiş ve çalışmada gerçekleştirilen deneyler sonucunda %87.2 genel sınıflandırma doğruluğu elde etmiştir.

Denetimli öğrenme yöntemlerinden bir diğeri olan Derin Öğrenmeyi kullanan VoxNet, Maturana & Scherer (2015) tarafından geliştirilmiştir. Hacimsel doluluk tahminini temsil eden hacimsel bir grid ve doğrudan bu gridden sınıf etiketini öngören bir 3B Evrişimli Sinir Ağları (Convolutional Neural Networks – CNN) olmak üzere iki ana bileşenden oluşur. Nokta bulutunu ve nokta bulutuna ait ek özellikleri kullanarak gerçek zamanlı obje tanıma yapabilen etkili bir yaklaşımdır.

Hu % Yuan (2016), hava Lidar nokta bulutundan Sayısal Arazi Modeli (SAM) çıkarımı için Derin Öğrenme temelli bir sınıflandırma yöntemi geliştirmiştir. Bu yönteme göre her bir nokta için bir pencere içerisindeki komşu noktalar çıkartılmıştır ve bu noktalar sigmoid fonksiyonu yardımıyla RGB değeri olan piksellere eşlenmiştir. CNN model, zemin veya zemin olmayan olarak etiketlendirilen noktaların görüntüsünü girdi olarak kullanmıştır. ISPRS veri seti için Tip-1 hatası %0.67, Tip-2 hatası %2.262 ve toplam hatası %1.22 olan yöntemin, kırk taneden fazla karmaşık alan için Tip-1 hatası %3.6, Tip-2 hatası %2.2 ve toplam hatası %2.9 olarak hesaplanmıştır.

PointNet ve PointNet++, Qi vd. (2017a, 2017b)'nin geliştirdiği, noktanın nokta özelliğini kaybetmeden sınıf etiketlerini aldığı 3B CNN yöntemidir. PointNet'te girdi olarak direkt nokta bulutu halindeki objeler kullanılırken, PointNet++'ta nokta bulutu komşuluk ilişkisi olan noktaların yerel alt yığınlara ayrılmış hali kullanılmaktadır. Çıktıda hem noktalar bir etiket değeri almakta; hem de segmantasyon gerçeklemektedir.

Politz & Sester (2018), tam evrişimli bir kodlayıcı-kod çözücü model oluşturmuşlardır. Modelin girdisi için hava Lidar ve yoğun görüntü eşleştirme (dense image matching - DIM) nokta bulutlarının 2B görüntü tarama düzlemine yansıtılıp her bir tarama hücresi için minimum, maksimum ve ortalama yükseklik değerleri hesaplanmıştır. Daha sonra zemin, zemin olmayan, bina ve veri-yok olarak kategorize edilen veri setinin genel sınıflandırma doğruluğu hava Lidar verisi için %96, yoğun görüntü eşleştirme verisi içinse %83 olarak elde edilmiştir. Politz vd. (2018), lazer tarama verileriyle hem nokta temelli semantik sınıflandırma, hem de nesne temelli sınıflandırma yapmak için CNN kullanmışlardır. Nokta temelli eğitimin sonucunda nokta bulutu zemin, zemin olmayan ve araba etiketlerine göre sınıflandırılmıştır ve test sırasında %91.14'lük bir genel doğruluk elde edilmiştir. Nesne tabanlı sınıflandırma sonucunda ise genel doğruluk %75 olarak hesaplanmıştır.

Zhou & Gong (2018), gerçekleştirdikleri çalışmada derin sinir ağları ile hava Lidar verisinden yerleşim yapılarını %93 sınıflandırma doğruluğuyla tespit etmişlerdir. Önceden tanımlanmış geometrik veya doku detaylarını gerektirmeksizin, çeşitli

nokta yoğunluklu ve hasarlı bina objeleri içeren hava Lidar veri setlerine uygulanabilir bir yaklaşım önermişlerdir.

Winiwarter vd. (2019), nokta bulutunu Öklid uzayı benzerliği ile yığınlara bölerek modelin girdisini oluşturmuştur. ALSNet adı verilen bu ağ, PointNet'in biraz daha geliştirilmiş bir yaklaşım olarak öne sürülmüştür. Dört test verisinden, bir şehir alanı için %95.8, ormanlar ve daha küçük yerleşim yerlerine sahip iki alan için % 82.6 ve % 86.7, yüksek dağ arazisindeki bir alan için ise % 63.6 genel doğrulukları elde edilmiştir.

Yapılan çalışmalar, hava Lidar verisinin işlenmesi ve sınıflandırılmasında yapay zeka yöntemlerinin her geçen gün daha fazla katkı sağladığını göstermektedir. Elle yapılan ve zaman alan eski yöntemlerin yavaş yavaş yerini yapay zeka yöntemlerine bıraktığı gözlenmiştir.

1.2 Tezin Amacı

Tezin amacı, günümüzde merak uyandıran konulardan biri olan yapay zeka yöntemlerinin, hava Lidar verilerine ne kadar sürede ve hangi doğrulukta sınıflandırma ve filtreleme yapabildiğini araştırmak ve kıyaslamaktır.

Bu çalışmada, kırsal alanlarda hava Lidar verilerinin denetimli ve denetimsiz yapay sinir ağları ve makine öğrenmesi yöntemleriyle hem sınıflandırma hem de filtreleme performansları değerlendirilmiştir. Nokta bulutunun x, y koordinat ve z yükseklik bilgisiyle birlikte yansıma değeri (intensity) de kullanılarak, denetimsiz yapay sinir ağlarından Kendini Düzenleyen Haritalar yöntemi, denetimli yapay sinir ağlarından Derin Öğrenme ve makine öğrenmesi yöntemlerinden Rastgele Orman algoritması iki farklı kırsal alanda test edilmiştir.

1.3 Hipotez

Hava Lidar verileri, yapay zeka yöntemleriyle yüksek doğrulukla denetimli ve denetimsiz olarak sınıflandırılabilir ve filtrelenebilir.

Denetimsiz yapay sinir ağları yöntemlerinden Kendini Düzenleyen Haritalar için Lidar verisinin yükseklik değeri (z) ve yansıma değeri (intensity)'nin ağırlıklarının

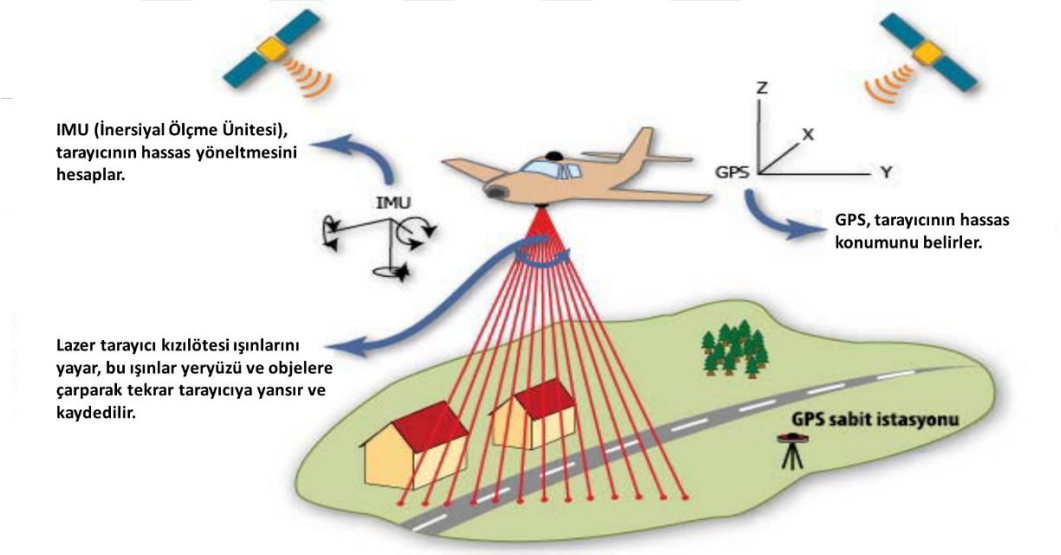
x, y koordinat değeri ağırlıklarına göre daha yüksek alınması sınıflandırma ve filtreleme sonuçlarının doğruluğunu arttırabilir.

Denetimli yapay sinir ağları yöntemlerinden Derin Öğrenme için x, y koordinat değerlerini normalize edip, yükseklik değerini ve yansıma değerini normalize etmeden kullanmak sınıflandırma ve filtreleme sonuçlarının doğruluğunu arttırabilir. Rastgele Orman algoritmasında ise veriler normalize edilmeden de iyi bir sınıflandırma doğruluğu elde edilebilir.

2
HAVA LİDAR VERİSİ

2.1 Hava Lidar Verilerinin Elde Edilişi

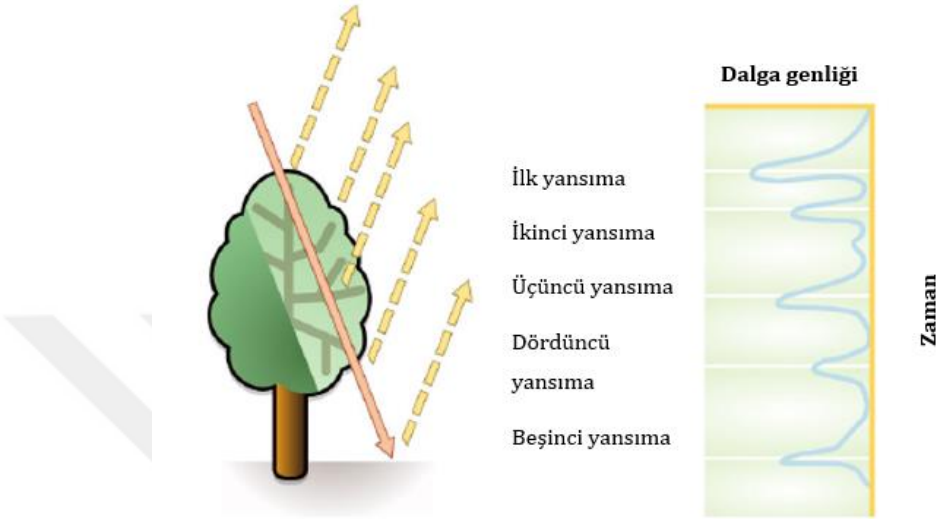
Hava Lidar lazer, GPS, IMU gibi birden fazla bileşen içerebilen aktif bir uzaktan algılama sistemidir. Uçak veya helikopterin yanı sıra drone gibi teknolojik hava araçlarına da yerleştirilebilen Lidar sistemleri, istenilen veri kalitesine uygun hazırlanacak bir uçuş planı ile yüzeye ait 3B konum verisi toplar. Temel çalışma prensibi, lazer pulsunun gönderildiği an ile yansımanın yüzeye çarpıp kaynağa geri ulaştığı an arasındaki zaman farkının hesaplanmasına dayanır. Bu zaman farkı ikiye bölünür ve ışık hızı ile çarpılarak yüzeye olan mesafe elde edilir.



Şekil 2.1 Hava Lidar sisteminin çalışma prensibi (URL1)

2.2 Hava Lidar Veri Yapısı

Hava Lidar ile x, y, z koordinatları yanı sıra yansıma değeri (intensity) gibi öznitelikleri de içerebilen 3B nokta bulutu elde edilir. Lidar sistemlerinin en önemli özelliklerinden biri çoklu yansıma özelliğidir.



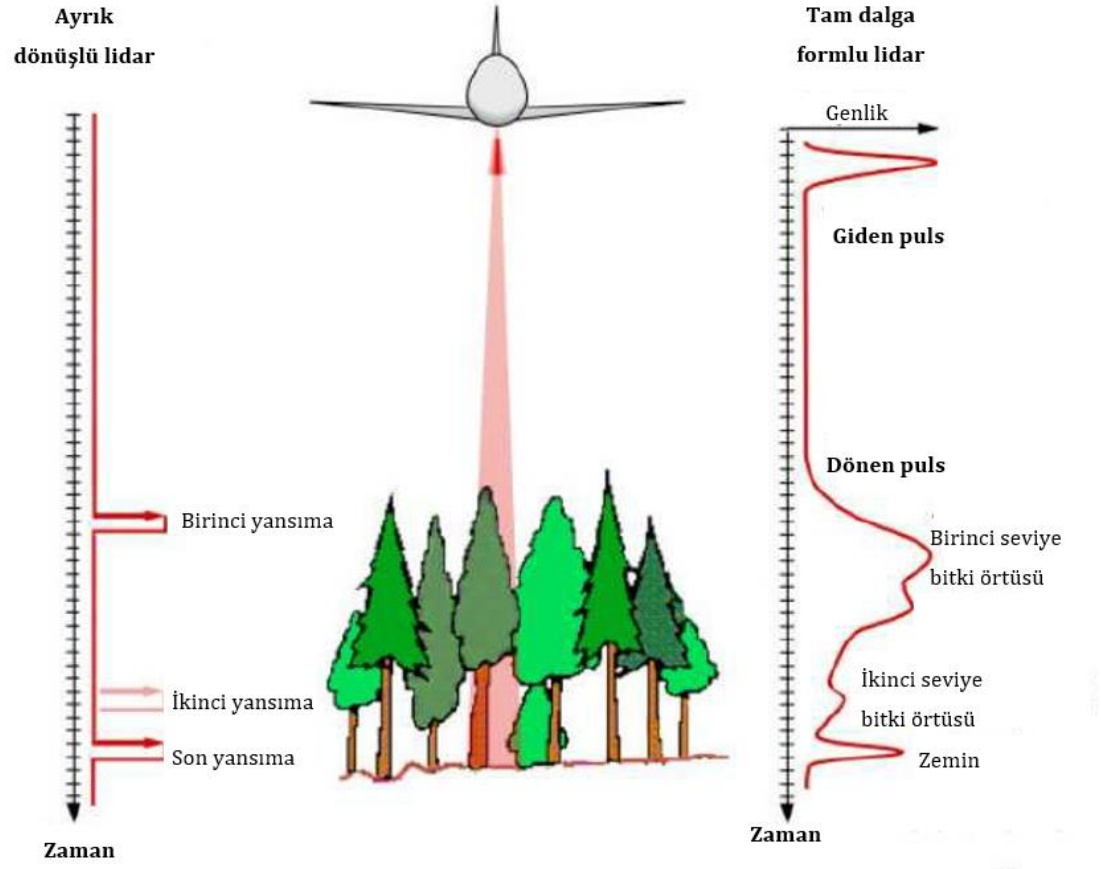
Şekil 2.2 Hava Lidarın çoklu yansıma özelliği (URL2)

Lidar sistemleri, geri dönüş sinyalinin kaydedilme şekline göre ayrık dönüşlü sistemler ve tam dalga formlu sistemler olarak sınıflandırılabilir.

Tipik ayrık dönüşlü sistemler, yaklaşık 3 m veya daha fazla mesafeye sahip olmaları durumunda iki dönüş sinyalini ayırt edip iki farklı sinyal olarak kayıt edebilmektedir. Yakalanan yükseklik özelliğine ek olarak, her gidiş ve dönüş sinyali, sinyallerin sayısı ve yoğunluğu gibi diğer nitelikleri de yakalar. Bu ek nitelikler, Lidar verilerinin analizinde kullanılabilir. Tam dalga boyu formuna dayalı sınıflandırma ise, bir dönüş sinyalinin tam profilini sabit zaman aralıklarında örnekleyerek kullanır. Daha ayrıntılı bir şekilde ağaç ve orman özellikleri araştırılmak isteniyorsa tam dalga formlu Lidar verileri kullanılmaktadır. Tam dalga formlu Lidar verileri, özellikle ormanın altındaki genç bitki örtüsünün tespitinde oldukça kullanışlıdır (Reitberger vd., 2008a).

Nokta sınıflandırma algoritmaları, veri tipi kaynağına bağlı olarak nokta temelli sınıflandırma, tam dalga boyu temelli sınıflandırma ve veri füzyonu temelli

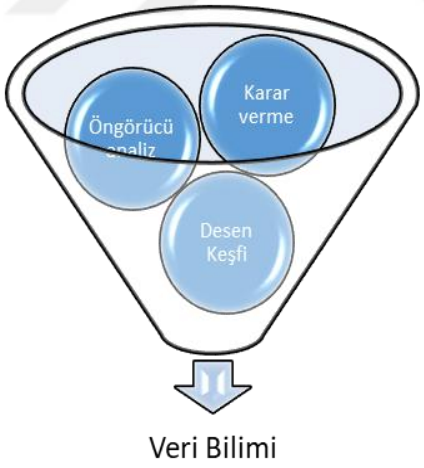
sınıflandırma olmak üzere üç ana gruba ayrılır. Nokta tabanlı sınıflandırma teknikleri genellikle ayrık dönüşlü Lidar verilerini işlemede kullanılmaktadır. Tam dalga boyu formlu Lidar verilerinin, geleneksel ayrık-dönüşlü nokta bulutu verisine kıyasla hacmi çok büyüktür. Bu nedenle ayrık dönüşlü Lidar sistemleri, işleme ve/veya görüntü işleme ile veri füzyonu tabanlı yöntemlere göre hızlı veri işleme sağlamaktadır. (Zhang, Lin, & Ning, 2013).



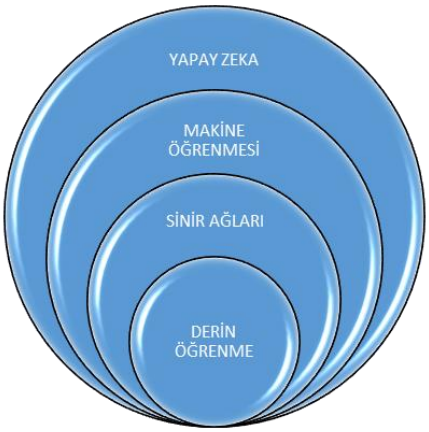
Şekil 2.3 Ayrık dönüşlü Lidar ve tam dalga formlu Lidar (Ferraz vd., 2009)

3
VERİ BİLİMİNE GİRİŞ VE YAPAY ZEKA

İnternet sayesinde kuruluşların ve işletmelerin verilerinin büyümesiyle bir veri seli yaşanmıştır. Bilgiyi anlayabilmek, analiz edebilmek ve bu bilgilerden çıkarımlarda bulunarak sonuçlar elde edebilmek amacıyla tıp, mühendislik, sosyal bilimler gibi pek çok alanda **büyük veri (big data)** kullanımı yaygınlaşmıştır. Sürücüsüz arabalar ve araçlar, yüz tanıma sistemleri, dünya satranç şampiyonunu yenen bilgisayarlar, hava durumu tahminleri veri biliminin önemli katkılar verdiği çalışmalardır. Veri biliminin Karar Verme (Desicion Making), Öngörücü Analiz (Predective Analysis) ve Örüntü Keşfi (Pattern Discovery) olmak üzere esas 3 bileşeni vardır (Gabriel, 2019). Bu üç temeli gerçekleştirebilmek için **Yapay Zeka**'ya ihtiyaç vardır. Yapay Zekâ, geniş anlamda makinelerden beklediğimiz veya hayal ettiğimiz insan zekasının bir çeşit temsili gösterimidir.



Şekil 3.1 Veri bilimi bileşenleri



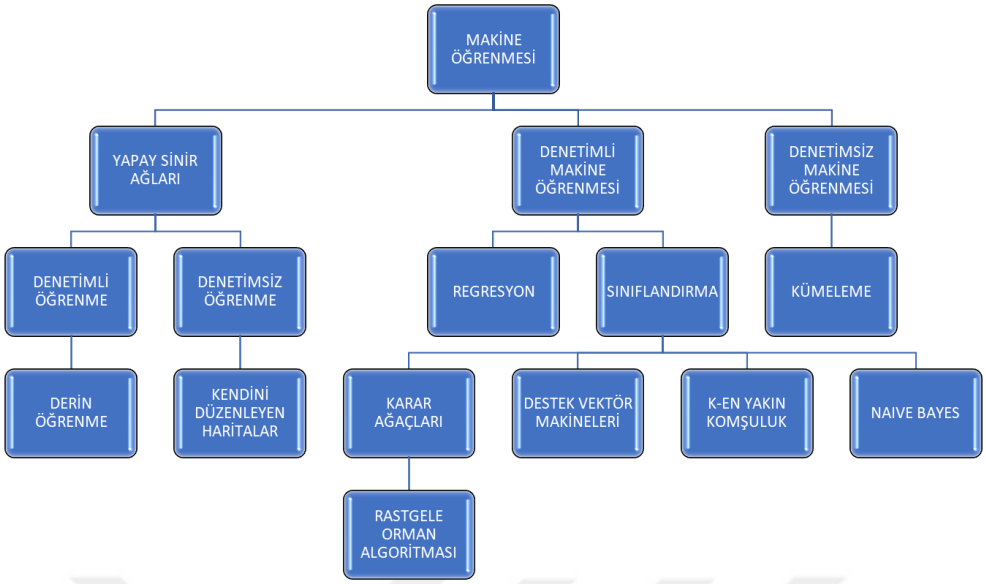
Şekil 3.2 Sık kullanılan yapay zeka yöntemlerinin birbirleriyle ilişkisi (Sotoudeh vd., 2019)

3.1 Makine Öğrenmesine Giriş

Makine öğreniminin öncüsü Arthur Samuel, makine öğrenimini "açıkça programlanmadan bilgisayarlara öğrenme yeteneği veren bir çalışma alanı" olarak tanımlamıştır. Makine öğrenimi, daha önce eğitim verilerinden öğrenilen özelliklere dayalı olarak sınıflandırma ve tahmine odaklanır (Buzcak & Güven, 2016).

Makine öğrenmesi, kullanılan veri setinin etiketli olup olmamasına göre denetimli ve denetimsiz öğrenme olmak üzere ikiye ayrılır. **Denetimli öğrenme**de veri seti eğitim ve test olmak üzere ikiye ayrılır. Örneğin, bir nokta bulutu içerisinde noktaların sınıf etiketi varsa, sınıf etiketli noktalar eğitim verisi olarak kullanılır. Sınıf etiketli eğitim verisi kullanılarak diğer noktaların (test verisinin) sınıf etiketleri tahmin edilir. Bir makineyi gerekli doğruluk seviyesinde eğitmek, algoritmayı önemli miktarda veri ile beslemeyi gerektirir. Algoritma veya model, beslendiği eğitim verilerini öğrenirken kendini düzenler.

Denetimsiz öğrenmede ise sınıf etiketleri bulunmadığı için veri setinin eğitim ve test olmak üzere ikiye ayrılmasına gerek kalmaz. Örneğin, sınıf etiketleri bulunmayan nokta bulutu içerisinde birbirine benzer noktaları kendi içerisinde gruplar, daha sonra bu gruplardan kümeleri oluşturur.



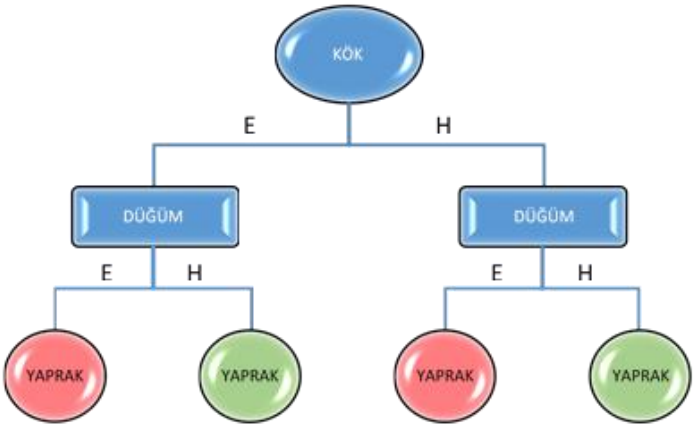
Şekil 3.3 Makine öğrenmesinin alt dalları içerisinde derin öğrenme, kendini düzenleyen haritalar ve rastgele orman algoritmasının yeri (Wang vd., 2020; Xue vd., 2019)

Wisdom of Crowds (Kalabalıkların Bilgeliği) yaklaşımına göre büyük bir topluluğun görüşü tek bir bireyin (uzman bile olsa) görüşünden daha doğru olabilir. Makine öğrenmesinde topluluk yöntemlerinde de durum böyledir. Birçok veri bilimcisi ve makine öğrenimi geliştiricisi, aslında farklı yöntemlerin sonuçlarının ortalamasını alan toplu yöntemler (yöntemlerin bir kombinasyonu) kullanır, ayrıca bu şekilde amaçlarına ve uygulamalarına en uygun yöntemi seçerler (Gabriel, 2019).

3.2 Rastgele Orman Algoritması (Random Forest Algorithm)

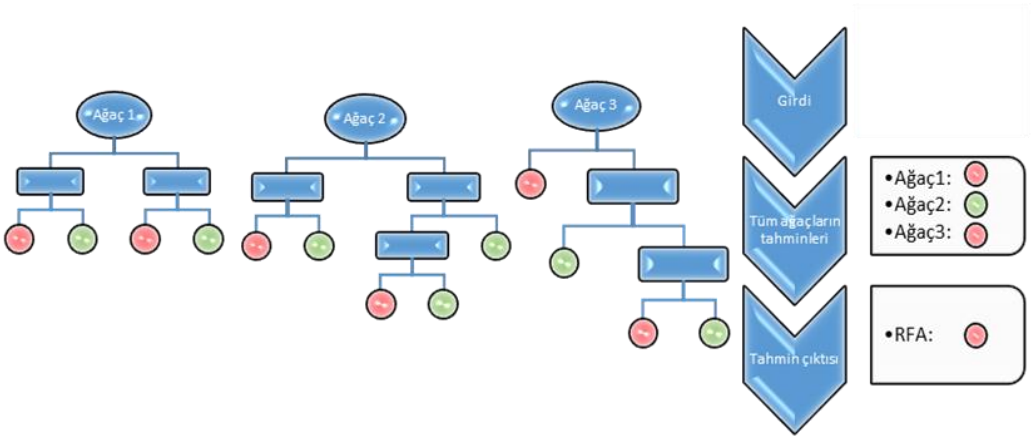
Karar ağaçları hem sınıflandırma hem de regrasyon analizlerinde kullanılabilen bir denetimli makine öğrenmesi yöntemidir. Karar ağacında yer alan her alan düğüm (node) olarak isimlendirilir. İlk düğüm kök (root), sonrakiler ara düğümler (interval nodes), en sonda yer alan ve nihai kararı belirten düğümler ise yaprak (leaves) olarak adlandırılır. Karar ağacı veri setini karar alanlarına yani alt düğümlere bölerek çalışır. Hata fonksiyonunu minimize etmek için sınıflandırma problemlerinde genellikle Gini ve Entropi fonksiyonları kullanılır. Karar ağaçlarının

en büyük dezavantajı aşırı öğrenme (overfitting)'dir ve karar ağacı çok büyük olduğu zaman yorumlamak zorlaşır (Arslan, 2020).



Şekil 3.4 Bir karar ağacının yapısı (Sa vd., 2016)

Rastgele Orman (RO) algoritması ise makine öğrenmesinin karar ağaçları yönteminin bir koleksiyonudur. Veri bilimi ve makine öğrenmesi söz konusu olduğunda, toplu bir yaklaşımın daha iyi sonuçlara yol açabileceğinden önceki bölümde bahsedilmişti. Karar ağaçları ve rastgele orman algoritması için de durum böyledir. Rastgele orman algoritması, rastgele birden çok karar ağacını oluşturur ve "Rastgelelik faktörü" her bir karar ağacındaki hataları ve eşiği düzeltmeye yardımcı olur (Adam, 2016). Daha sonra her bir ağacın tahmini birleştirilerek daha doğru ve istikrarlı bir tahmin sonucu üretilir. Ayrıca bu yöntem ile aşırı öğrenme sorunu büyük ölçüde önlenir.



Şekil 3.5 Rastgele orman algoritmasının çalışma şekli (Fu, 2017)

Rastgele örnekleme (bootstrap), metodunun esası, mevcut veri setinden çok büyük veri setleri üretmek üzere yeniden örnekleme yapmaktır. Böylece küçük sayıdaki veri setleri için de çeşitli istatistikler belirlenebilmektedir. Bootstrap metodu güven aralıkları, hipotez testi ve regresyon analizinde kullanılabilmektedir (Takma & Atıl, 2016)

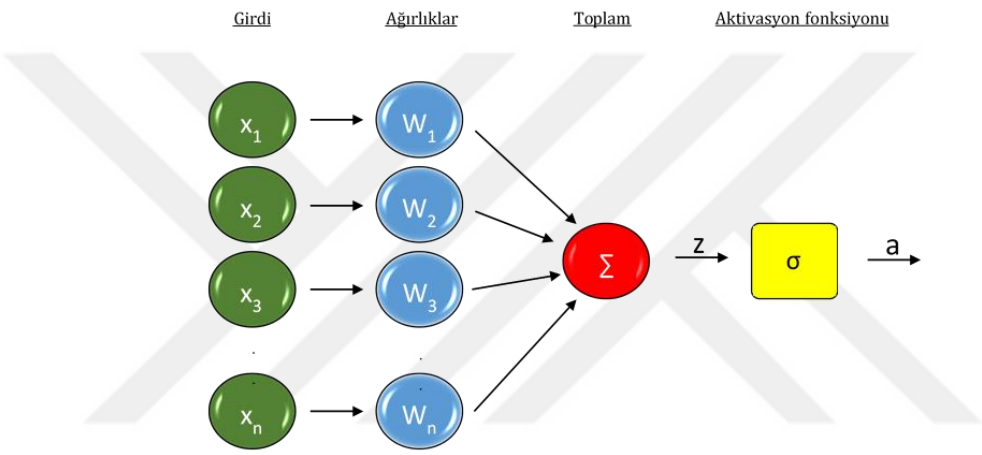
RO algoritmasında birden fazla karar ağacı olduğu için Torbalama (Bagging) yani toplu rastgele örnekleme yapılır. Değiştirmeli rastgele örnekleme (bootstrap with replacement) kullanılarak her bir karar ağacı farklı örneklem verileri ile eğitilir. Hatta bazen bir veri bir torba içerisine birden çok kez seçilebilir veya hiç seçilemeyebilir.

Tahmin sonucu tüm karar ağaçlarından çıkan sonucun ortalamasıdır. Tahmin doğruluğu da tüm karar ağaçlarının doğruluğunun ortalamasıyla ölçülür. Bu nedenle karar ağacında kullanılan bir özelliğin tahmin üzerindeki kısmi önemini ölçmek çok kolaydır.

RO algoritması ile model geliştirebilmek pek çok yönteme görece daha az zaman alır ve hızlı performans gösterir. Varsayılan hiperparametreler genellikle iyi bir tahmin sonucu oluşturduğu için kullanışlı bir algoritma olarak kabul edilir. Hiperparametre sayısı az ve anlaşılması kolaydır.

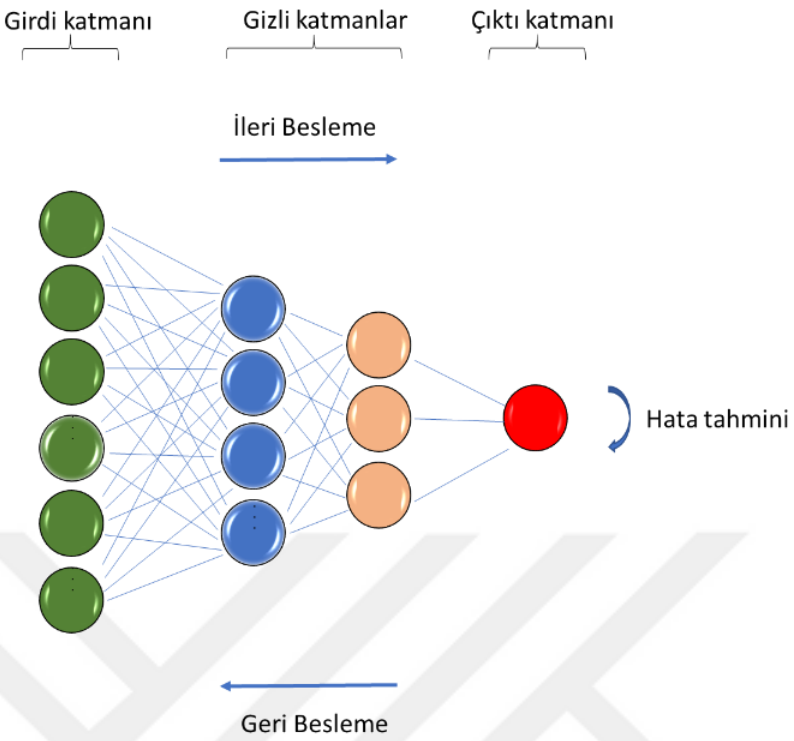
3.3 Yapay Sinir Ağlarına Giriş

Yapay Sinir Ağları (YSA), karmaşık görevlerin nöron adı verilen bir dizi bağlantılı bileşenin birleşik eylemiyle gerçekleştirildiği Yapay Zekaya (Artificial Intelligence - AI) biyolojik olarak ilham veren bir yaklaşımdır. 1958'de Rosenblatt ileri beslemeli tek katmanlı algılayıcıyı (Single Layer Perceptron) önermiştir. Şekil 3.6'da, hesaplama birimleri veya nöronlar, sinaptik bağlantıları temsil eden eğitilebilir ağırlıklar aracılığıyla birbirine bağlanan dairelerle temsil edilmektedir. Bu tek katmanlı algılayıcı, yalnızca doğrusal olarak ayrılabilir sorunları çözebilmiştir (Camuñas-mesa vd., 2019).



Şekil 3.6 Tek katmanlı algılayıcı

1960'ların sonlarında, doğrusallık sınırlamasının üstesinden gelmek için çok katmanlı algılayıcılı (multi layer perceptron) sinir ağları önerildi. Çok katmanlı ileri beslemeli ağ, katmanlar halinde düzenlenmiş nöronlardan oluşur ve nöronların yalnızca bir yönde (ileri) bir bağlantısı vardır. Bir girdi katmanı, bir çıktı katmanı ve bir veya daha fazla gizli nöron katmanından meydana gelir. Tüm katmanlarda büyük ölçüde benzer bir hesaplama gerçekleşir. Çok katmanlı ileri beslemeli ve geri yayılımlı ağlarda ise temel fikir çıktı katmanından elde edilen hatanın gizli katmana geri yayılması ve gizli katman ile giriş katmanı arasındaki ağırlıkların eğitimine rehberlik etmek için kullanılmasıdır (Wang ve Fu, 2009). Ağırlık, eşik gibi parametreler ile aktivasyon fonksiyonları iteratif olarak görev alır.



Şekil 3.7 Çok katmanlı ileri beslemeli geri yayılımlı yapay sinir ağları

3.3.1 YSA Parametreleri

Girdi birimi, ağırlıklı girdileri alır ve bir eşik ile birlikte alınan girdileri basitçe toplama birimine doğru besler. Algılayıcı, aşağıdaki gibi bir doğrusal fonksiyon ile ifade edilir:

$$y = wx + b \tag{3.1}$$

Denklemdeki x girdi değeridir, y ise girdiye ait skoru veren, x'e bağlı değişkendir. w ağırlık parametresi, b ise eşik değeridir. Toplam biriminde her girdinin skoru aşağıdaki denklem ile bulur (Krose & Smagt 1996).

$$\sum_{i=1}^{n} w_i x_i + b \tag{3.2}$$

Yapay sinir ağlarında eğitime başlarken ilk ağırlıkları rastgele vermek iyi sonuçlar vermeyebilir. Eğitim sürecinde ağırlıkları düzeltmek ve sonra yeniden düzeltmek için pek çok iterasyonun ardından yüksek doğruluk elde edilir. Başka bir deyişle, eğitim, **Ağırlık** ve **Eşiğin** optimum değerlerini bulma sürecidir. Bu parametreler **e**ğitim boyunca güncellenir.

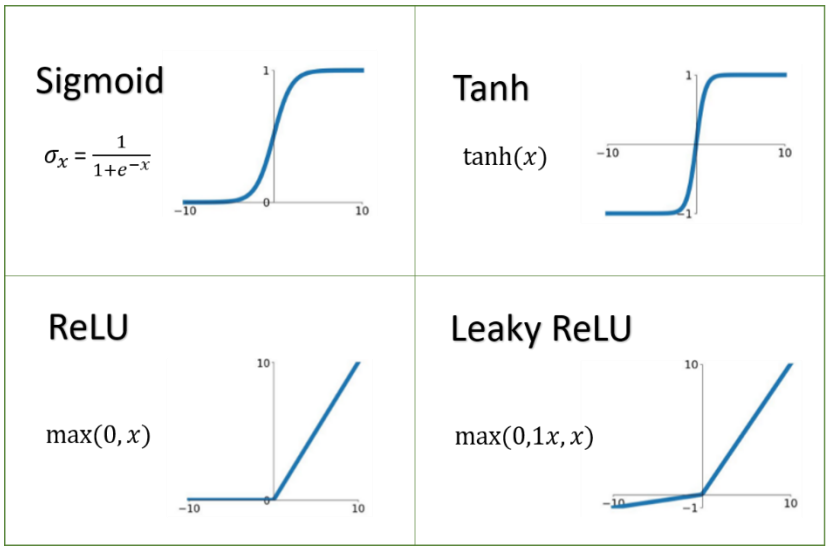
Ağırlık, girdilerden çıktılara olan etkileri belirler. Pozitif veya negatif pek çok ağırlık, çeşitli girdi verilerle birlikte çalışarak istenilen çıktı verileri üretir.

Eşik (Bias), genellikle ağı eğitmek için kullanılan verilere özgü önyargıyı ve favoriliği yansıtır. Sinir ağının bir düğümünün başlama kolaylığını belirler. Düğümün sonucu büyük ölçüde eşik parametresiyle belirlenir. Bu yüzden ağın oldukça önemli bir parametresidir.

3.3.2 Aktivasyon Fonksiyonları

Aktivasyon fonksiyonu, ağırlık ve eşik değerleriyle hesaplanan bilgilerin sonraki nörona iletilip iletmeyeceğine karar verir. En basit YSA olan tek katmanlı algılayıcı için bu durumda ikinci bir katman olmadığından, gelen sonuç eşik değerden büyükse nöron aktive olur. Bu nedenle tek katmanlı algılayıcı, ikili bir sınıflandırıcı olarak işlev görür.

Aktivasyon fonksiyonu, doğrusal olmayan çok katmanlı ağların da öğrenmesine önemli ölçüde yardımcı olur. Bazı aktivasyon fonksiyonlarına Relu0, Sigmoid, Tanh, Leaky Relu örnek olarak verilebilir. Aktivasyon fonksiyonu seçilirken doğrusal olmayan fonksiyonlar seçilir ve geri beslemeli ağlarda hesaplamayı kolaylaştırabilmek için türevi kolay alınan fonksiyonlar tercih edilir (Buduma & Locascio, 2017).



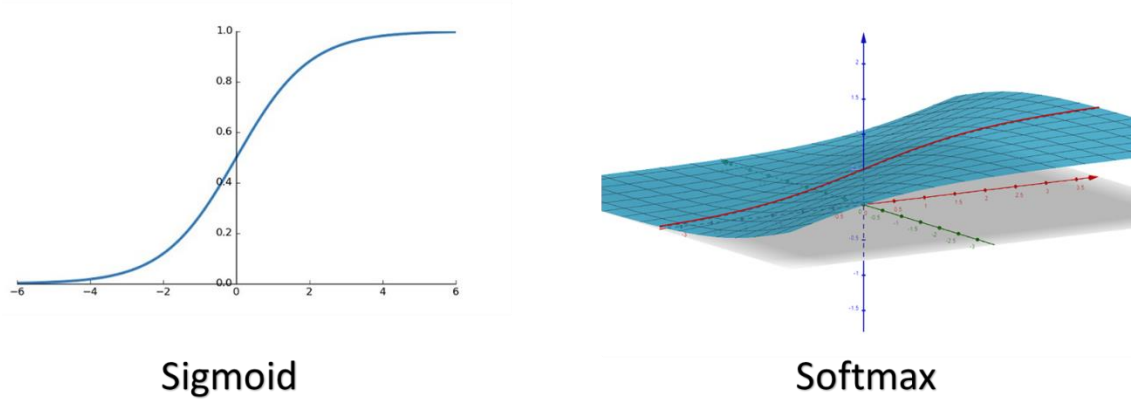
Şekil 3.8 Aktivasyon fonksiyonları

Softmax aktivasyon fonksiyonu da Sigmoid gibi girdi verilerinin dağılımını [0, 1] arasına çeker. Bir *K* gerçek sayı kümesini 1'e ulaşacak şekilde bir olasılık dağılımına normalleştirir. İkili sınıflandırma problemleri için genelde Sigmoid fonksiyonu kullanılırken, çoklu sınıflandırma problemleri için Softmax fonksiyonu kullanılır.

İkiden fazla girdiye sahip bir Softmax fonksiyonunu gerçek anlamda çizmek mümkün olmasa da ana fikri Sigmoid ile aynıdır: Girdi verisindeki diğer değerlerinin ne kadar büyük veya küçük olduğuna bağlı olarak sıfır noktası kayan bir sigmoid gibi düşünülebilir. Softmax foksiyonunda önemli olan girdilerin gerçek değerleri değil, girdilerin arasındaki mesafeleridir. Başka bir deyişle, tüm değerler bir miktar sabit kaydırılırsa, bunun bir önemi olmaz. Softmax fonksiyonu Eşitlik 3.3'teki gibi ifade edilir:

$$i = 1, \dots, K \; ve \; z=(z_i, \dots z_K) \in \mathbb{R}^K \; için$$

$$\sigma(z)_x = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.3}$$



Şekil 3.9 Sigmoid ve Softmax aktivasyon fonksiyonlarının kıyaslanması [URL3]

Şekil 3.9'daki grafiğin sağ tarafında, kırmızı eksen *X*, yeşil eksen *Y* ve mavi eksen Softmax'ın çıktısıdır. y = 0 için orijinal Sigmoid kırmızıyla ana hatları çizilmiştir. Daha büyük bir *y* için Sigmoid *x* ekseninin sağına kaydırılır, dolayısıyla daha büyük bir *x* değerine ihtiyaç duyulur. Daha küçük bir *y* için sola kaydırılırsa daha küçük bir *x* değeri ile aynı çıktı değerinde kalınabilir.

Softmax, derin öğrenme gibi karmaşık yapay sinir ağlarında olasılık dağılımını hesaplamak için kullanılan bir aktivasyon fonksiyonudur.

3.3.3 Kayıp (Loss) Fonksiyonu ve Optimizasyon

Yapay sinir ağının çıktısında tahmin edilen değerler ile gerçek değerler arasındaki farka **kayıp (loss) fonksiyonu** denir. Bir yapay sinir ağının performansı kayıp fonksiyonu ile ters orantılıdır. Hedef, kayıp fonksiyonunu minimize etmektir. Bunun için iteratif olarak her aşamada kayıp fonksiyonunu hesaplamak ve minimuma indirmek için **optimizasyon** yöntemleri uygulanır. İterasyon, performans istenilen seviyeye ulaşana kadar devam eder. *"Ortalama Kare Hata"*, YSA'da kullanılan bir kayıp fonksiyonudur. Matematiksel olarak Eşitlik 3.4'teki gibi tanımlanır. y_i'nin gerçek değeri ile y_i' tahmin değeri arasındaki kare farkların ortalamasıdır.

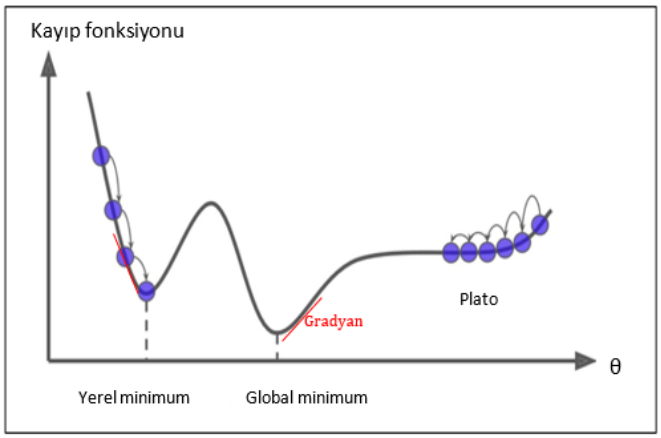
$$Ortalama\ Karesel\ Hata = \frac{\sum_{i=1}^{n}(y_i - y_i')^2}{n} \tag{3.4}$$

Gradyan optimizasyonu, kayıp fonksiyonunu değerini minimize etmek için kullanılan birinci dereceden iteratif bir algoritmadır. Bir fonksiyonun gradyanı anlık bir değişim oranı verir. Bir parametreye göre hatanın gradyanı hesaplanır ve hatanın türevinin negatifiyle orantılı olarak ağırlık değişir (Krose & Smagt 1996). Bu süreç tekrarlanarak en sonunda ilk pozisyona (YSA parametrelerinin başlangıç değerine) ve öğrenme hızına bağlı olarak Şekil 11'de görüldüğü gibi "yerel minimum", "global minimum" veya "plato" üzerine indirgenir. Gradyan temelli bazı optimizasyon araçları: *Momentum, Nesterov Accelerated Gradient, RMSProp, Adadelta* ve *Adam*.

Eşitlik 3.5'teki tahmin edilen y_i' değeri yerine aşağıdaki eşitlik yazılır:

$$y_i' = mx + c \tag{3.5}$$

Eşitlik 3.5'te m eğimi, c ise fonksiyonun x eksenini kestiği değeri gösterir. Fonksiyonun kısmi türevindeki m ve c değerleri fonksiyonun minimum değeri 0 olana kadar iteratif olarak güncellenir. Bu işlem "dışbükey" şekilli bir eğriyle sonuçlanır ve bu eğrinin en düşük noktası bulunur.



Şekil 3.10 Kayıp fonksiyonu, global ve yerel minimum, gradyan

3.4 Kohonen'in Kendini Düzenleyen Haritaları

Kohonen haritası olarak da bilinen KDH ilk defa Kohonen (1990) tarafından sunulmuştur. KDH kümeleme yöntemi veri izdüşümü ve nicelleştirmenin bir

kombinasyonu olarak düşünülebilir. Yarışmacı sinir ağları olarak, çok boyutlu öznitelik uzayında benzemezlikleri ölçer. Gizli katmanı olmayan, girdi ve çıktı katmanlarından oluşan denetimsiz yapay sinir ağlarını içerir. Her nöron, girdi verisi ile aynı boyutlu referans vektörü diye adlandırılan bir vektör ile ilişkilidir.

n boyutlu vektörleri içeren x giriş verisi:

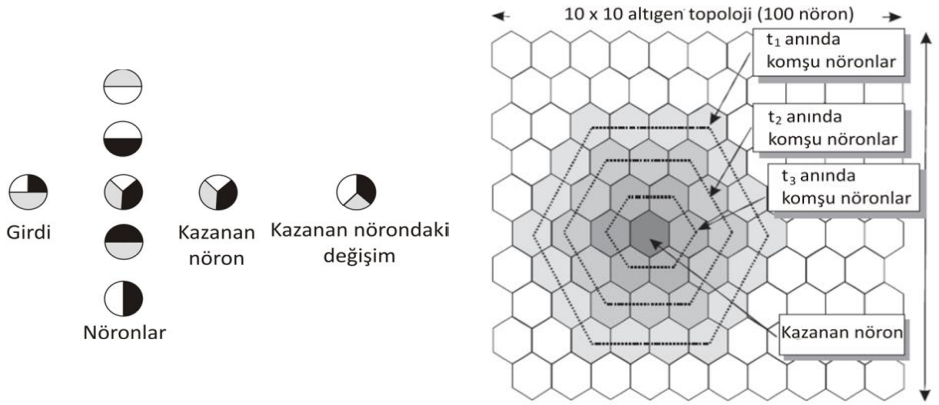
$$x = [\varepsilon_1, \dots, \varepsilon_n]T \in R^n \tag{3.6}$$

Burada; ε_i, x giriş vektörüne ait öznitelik değeridir. Her nöron bir m_i referans vektörünü içermektedir.

$$m_i = [\mu_1, \dots, \mu_n]T \in R^n \tag{3.7}$$

Burada; μ_i, m_i referans vektörüne ait rastgele öznitelik değeridir.

Öğrenme esnasında nöronlar birbirleri ile rekabet içindedir. Giriş vektörleri, Eşitlik 3.11'de görüldüğü gibi bir mesafe kriterine göre her bir referans vektörü ile karşılaştırılır. En küçük mesafe yani en büyük benzerliğe sahip nöron rekabeti kazanır. Rekabeti kazanan nöron ve komşuları her bir giriş vektörünü öğrenerek, referans vektörlerini günceller. Sonuç olarak, birbirine en çok benzer olan giriş vektörleri kendilerine en benzer nöronlara yerleşerek, hipotetik bir elastikiyete sahip harita ağı biçimindeki nöron ağını kararlı bir yapıya dönüştürür. Topolojik



Şekil 3.11 Kazanan vektörün girdi vektörüne göre değişimi (sol) ve eğitim boyunca azalan doğruluk yarıçapı (sağ) (Şen, 2013)

öğrenme biçimi sayesinde, geleneksel yöntemlere üstünlük sağlar (Kohonen, 2001; Yan & Thill, 2008).

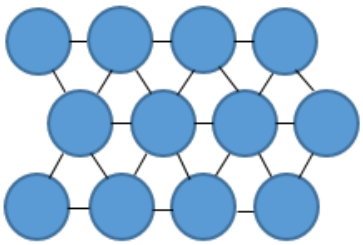
$$||x - m_c|| = min_i\{||x - m_i||\} \tag{3.8}$$

Burada; m_c, minimum mesafe veya maksimum benzerlik şartını sağlayan referans vektörüdür.

KDH sürecinde ilk adım ağın boyutunu (ağın x ve y yönünde kullanılan nöron sayısı) ve topolojisini belirlemektir. Nöron ağı, kare veya altıgen topolojiye sahip olabilir (Şekil 3.12). Öğrenme başlamadan, n-boyutlu referans vektörleri nöronlara genellikle rastgele olarak atanır. Eğitimi etkileyen temel parametrelerden biri; referans vektörlerini güncellemek için bir mesafe-ağırlıklı model olan komşuluk fonksiyonudur. Doğrusal ve Gauss modelleri en popüler fonksiyonlardır. Rekabeti kazanan nöron (en iyi eşleşen birim) ve komşuları Eşitlik 3.9 ile güncellenir.

$$m_i(t + 1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)] \tag{3.9}$$

Burada; t, iterasyon zamanını ve h_{ci}, komşuluk fonksiyonunu göstermektedir.



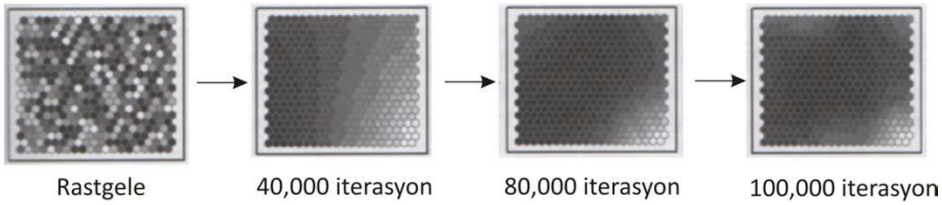
Şekil 3.12 Altıgen topolojiye sahip x-y yönlerinde nöron ağı

Gauss modeli:

$$h_{ci}(t) = \alpha(t)exp(-d_{ci}^2 / 2\sigma_i^2(t)) \tag{3.10}$$

Burada; d_{ci}, iyi eşleşen birim ile ilişkili komşu nöron arasındaki mesafeyi; $\alpha(t)$, t zamanda öğrenme katsayısını ve $\sigma(t)$, t zamanda kernel genişliğini göstermektedir.

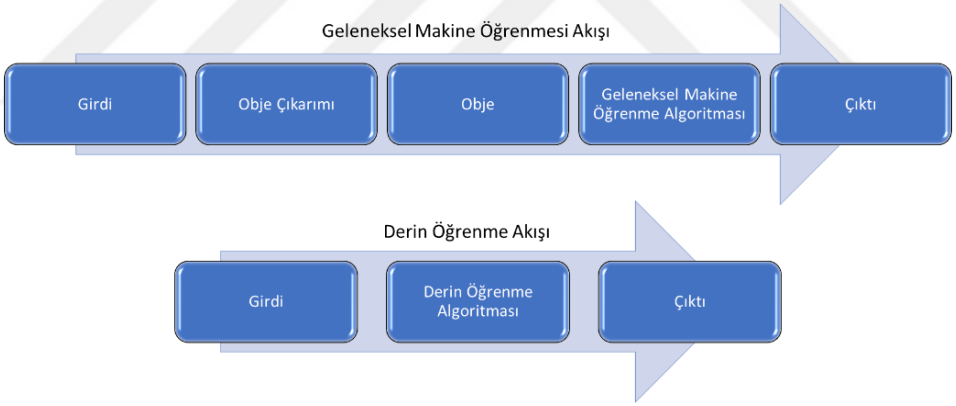
Gauss modelinde, komşuluk büyüklüğü kernel genişliği (σ) ile belirlenir ve sabit bir parametre değildir. Benzer biçimde, başlangıç öğrenme katsayısı (α_0) t zamanda yavaş yavaş azalan bir giriş parametresidir. KDH, tanımlanan iterasyon sayısı tamamlandığında kümeleme işlemine son verir (Skupin & Agarwal, 2008).



Şekil 3.13 KDH Eğitimi (Şen, 2013)

3.5 Derin Öğrenmeye Giriş

Derin Öğrenme, makine öğreniminin bir alt alanıdır. Ancak, insan beynindeki bağlantılı nöronlardan ilham alan Yapay Sinir Ağları alanında uzmanlaşmıştır. Ağ katmanlarının derinliği, derin öğrenmeye adını verir.



Şekil 3.14 Geleneksek makine öğrenmesi ve derin öğrenme arasındaki akış diyagramı farkı



Şekil 3.15 Derin öğrenmenin yaygın kullanılan bileşenleri

Yaygın olarak kullanılan derin öğrenme bileşenlerinden ağırlık ve eşik parametreleri, geri yayılım ve optimizasyon algoritmaları, kayıp fonksiyonu, aktivasyon fonksiyonları ve Softmax önceki bölümde anlatılmıştır. Derin öğrenmede kullanılan hiperparametreler; gizli katman ve gizli birim sayısı, öğrenme oranı (learning rate), yığın boyutu (batch size), iterasyon ve epoklardır (Gabriel, 2019). Hiperparametreler,

- Ağırlıkları ve eşiği kontrol eder,
- Ağırlık ve eşiğin son değerini belirlerler,
- Ağ tarafından öğrenilmezler.

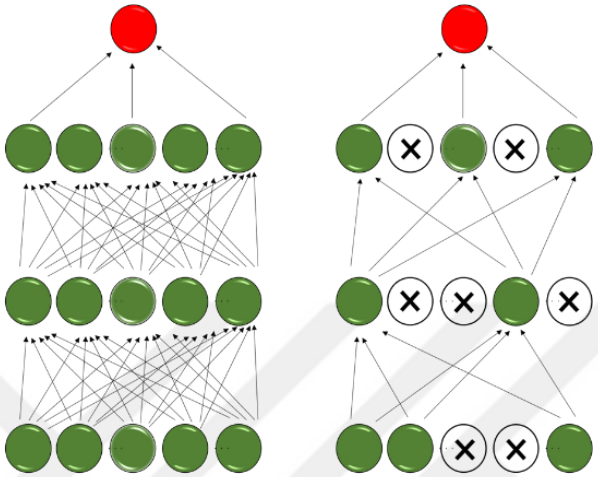
Gizli katman ve gizli birim sayısı, tüm sinir ağı mimarisine rehberlik eder. Genellikle veri setinin karmaşıklığına göre seçilir. Belirlenmesi oldukça kritik ve zordur.

Öğrenme oranı, öğrenme adımı büyüklüğü (step size)'nü belirler. Ağın eğitim hızı bu hiperparametre ile genellikle deneysel olarak belirlenir. Eğer yüksek öğrenme oranı isteniyorsa, optimizasyon sırasında atanan adım boyutu büyük seçilir ve eğitim daha hızlı biter; ancak bu doğruluğu da etkiler. Eğer öğrenme hızı çok yüksek olursa, optimizasyon aracı artık yakınsama yapmak yerine sapmaya başlar. Eğer düşük öğrenme hızı seçilirse, ağın eğitilmesi çok zaman alır, optimizasyon aracı global minimuma ulaşmak yerine yerel minimuma sıkışıp kalabilir.

Yığın boyutu (Batch size), eğitim sırasında ileri ve geri beslemede kullanılacak eğitim veri örneği sayısını belirler.

İterasyon, bir veri setindeki olabilecek yığın sayısıdır. Örneğin 1000 eğitim verisi için yığın sayısı 100 seçilirse, iterasyon sayısı 10 olur. Tüm iterasyonlar bittiği zaman ise bir **epok** tamamlanmış olur. Epok sayısı tüm eğitim setinin derin ağı eğitmek için kaç kez kullanılacağını belirler.

Seyrelteme (Dropout), derin öğrenmede aşırı uyum sorununa getirilen bir çözümdür. Bu yöntemde belli eşik değerin altındaki birimler geçici olarak seyreltilir, gelen ve giden bağlantılarıyla birlikte çıkarılır (Şekil 3.16). Böylece ağdaki zayıf bilgiler unutulur.



Şekil 3.16 Seyreltme

Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN)

Obje çıkarımı, görüntü sınıflandırma ve görüntü tanıma için en iyi derin öğrenme mimarisidir. Bir evrişimli sinir ağında her katmandaki nöronlar sadece öndeki katmanın belirli bir bölgesine bağlanır. Çıktı katmanını, girdi verilerinin olası çıktı sınıflarının her biri için bir sınıf skoru vektörüne indirger.

Bir evrişimli sinir ağı mimarisi 3 katmandan oluşur.

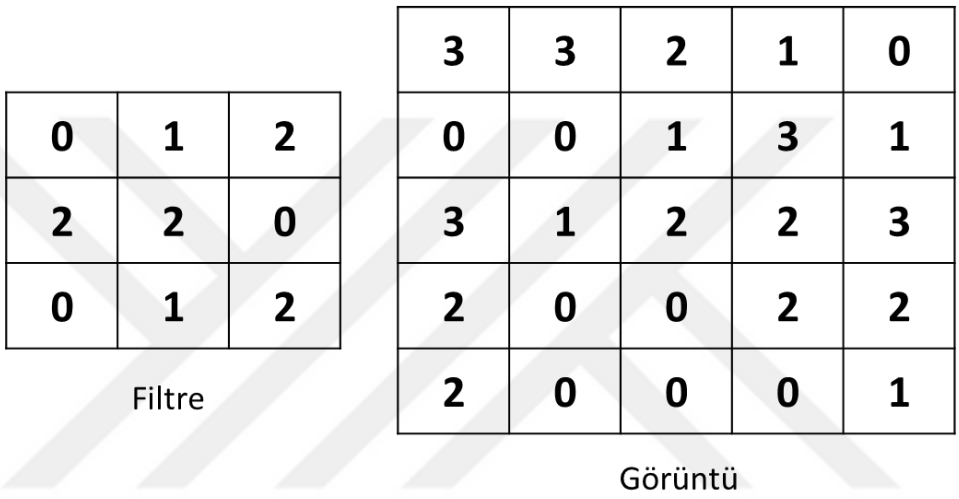
- Evrişim katmanı (Convolutional layer)
- Havuzlama Katmanı (Pooling layer)
- Tam bağlantılı katman (Fully connected layer)

Evrişim Katmanı

Evrişimli sinir ağı mimarisinin omurgasıdır. Evrişimli katmanın çıktı katmanı, nöronun bağlandığı girdi hacmindeki yerel bölge hakkında bilgiye sahiptir. Bu sayede girdi verilerinin küçük bölgelerinin özelliğini çıkarır.

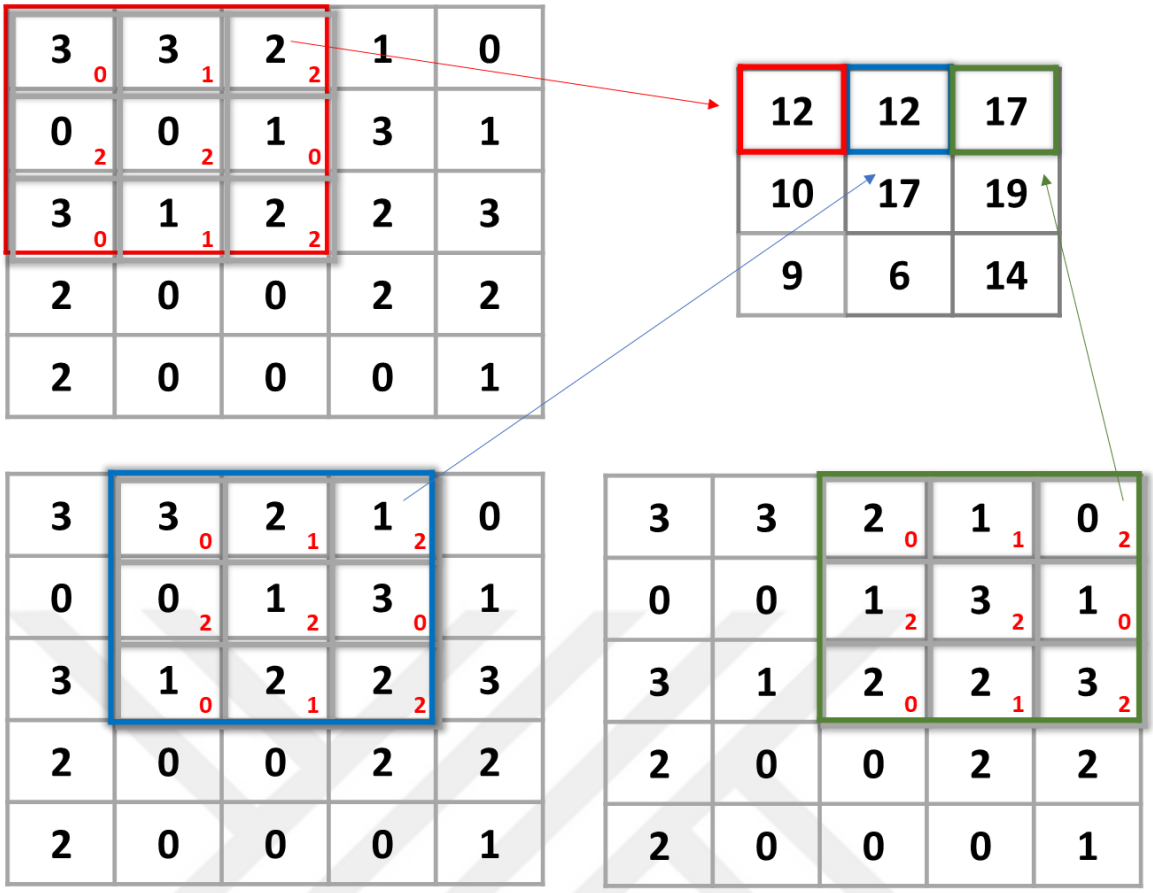
Evrişim işlemi (Convolution operation), evrişim girdi verilerini ve bir filtreyi içerir. "Filtrenin derinliği" kullanılacak filtre sayısını ifade eder. Bir evrişimden sonra filtrenin ne kadar (kaç piksel) hareket edeceği kademe (stride) ile ifade edilir. Dolgulama (padding) ise sınırın etrafına sıfır (0) eklemek yani doldurmak anlamına gelir.

Şekil 3.17'da 5x5'lik bir görüntüde evrişim işlemini gerçekleştirmek üzere, derinliği 1, kademesi 1 ve dolgulaması olmayan 3x3'lük bir filtre kullanılacaktır:



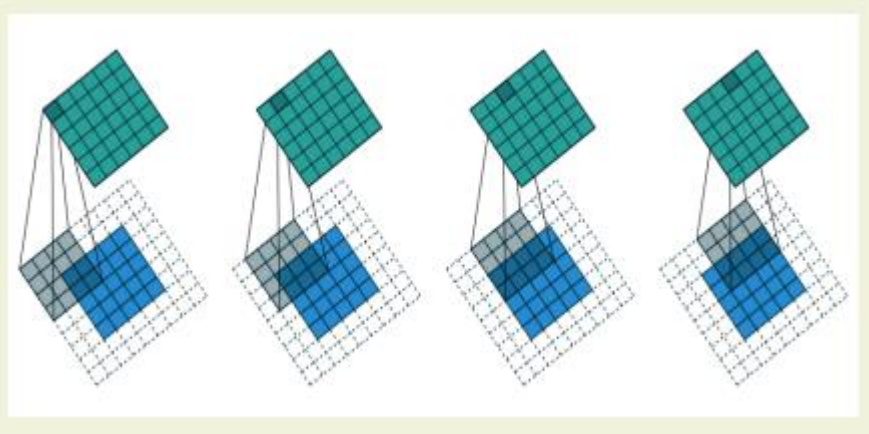
Şekil 3.17 Filtre ve görüntü: Solda 3x3'lük, derinliği 1, kademesi 1 ve dolgulaması olmayan filtre, sağda 5x5'lik görüntü

Evrişim işleminde, her bir piksel değeri karşılık geldiği filtrelerdeki piksel değerleriyle çarpılarak toplanır ve bu Şekil 3.20'deki gibi her adımda devam eder.



Şekil 3.18 CNN filtreleme işlemi

Evrişimden sonra 3x3 özellikli bir harita elde edilir. Dolgulama olmadığı için özellik haritasının boyutu girdi görüntünün boyutundan küçüktür. Şekil 3.18'de 5x5'te 3x3'e indirgenmiştir. Bu da dolgulama olmadan çok derin mimariler yaratılamayacağı anlamına gelir. Boyutu korumak ve özellik haritasındaki indirgemeyi azaltmak için sıfırlar eklenir. Şekil 3.19'da dolgulama ile filtreleme gösterilmiştir.



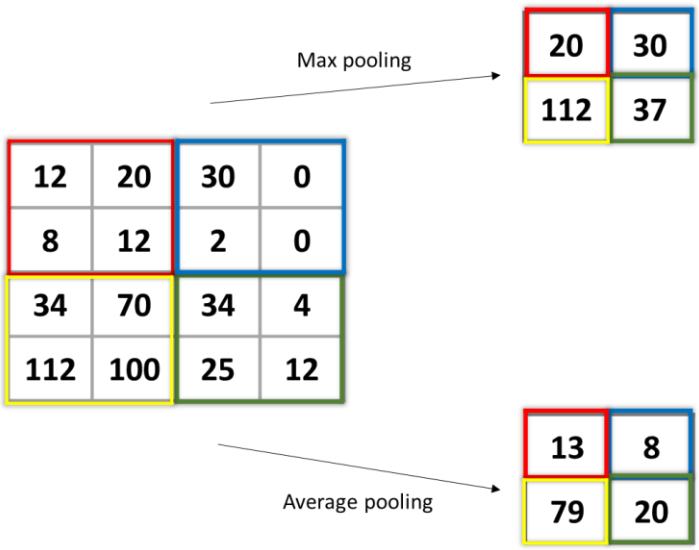
Şekil 3.19 Dolgulama ile CNN filtrelemesi

Evrişim katmanı, ilk katmanında basit özellikleri öğrenmeye meyillidir ve derinlik arttıkça özellikler karmaşıklaşır. İlk katmanda basit kenarları ve diğer basit özellikleri öğrenir.

Havuzlama Katmanı (Pooling Layer)

Mekansal boyutu küçültmek için art arda iki evrişim katmanı arasına bir havuzlama katmanı yerleştirilir. Havuzlama katmanı, parametreleri ölçeklendirir ve aynı zamanda aşırı öğrenmeyi engeller.

Maksimum Havuzlama (maximum pooling), filtre pencere boyutundaki en büyük değerin, **ortalama havuzlama (average pooling)** ise filtre pencere boyutundaki ortalama değerin dikkate alındığı havuzlama yöntemidir.



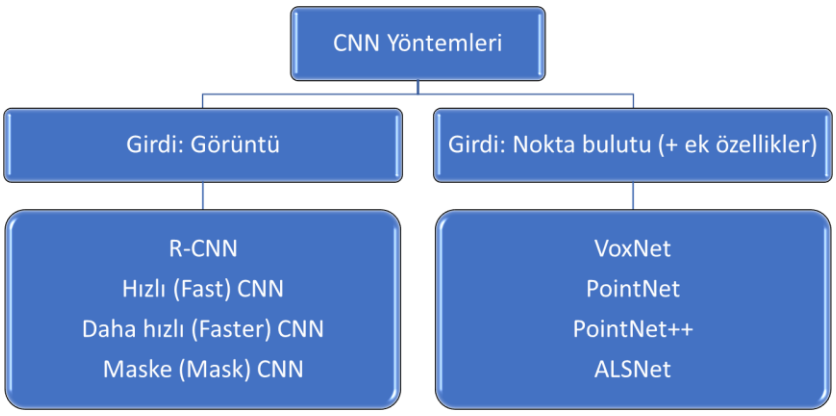
Şekil 3.20 Havuzlama yöntemleri

Tam Bağımlı Katman (Fully-Connected Layer)

Önceki katmandaki tüm aktivasyonlara bağlı nöronlara sahip düzenlenmiş ağlardır. Doğrusal olmayan verileri öğrenmede kolaylık sağlar.

3.5.1. CNN Yöntemleri

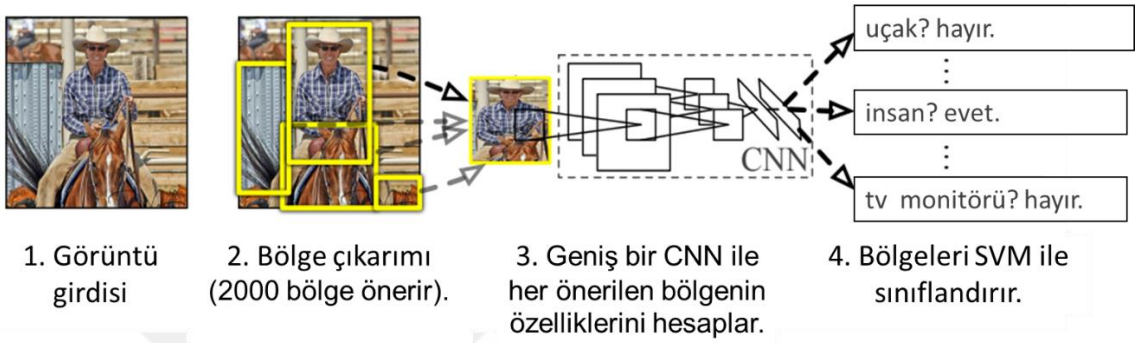
Günümüzde kullanımı yaygın olan CNN yöntemleri girdi verisine göre ikiye ayrılabilir:



Şekil 3.21 Nokta bulutu sınıflandırmada sıkça kullanılan CNN yöntemleri

3.5.1.1. Görüntü Temelli CNN Yöntemleri

Girshick vd. (2014), **R-CNN** modeli ile görüntülerden obje çıkarımı için üç modülden oluşan bir sistem sunmuştur. İlk modülde, kategoriden "seçici arama algoritması" ile bağımsız bölge önerileri oluşturulur ve bu öneriler, dedektör tarafından kullanılabilecek aday tespit kümesini tanımlar. İkinci modül, her bölgeden sabit uzunluklu bir özellik vektörü çıkaran büyük bir evrişimli sinir ağıdır. Üçüncü modül, sınıfa özgü doğrusal destek vektör makinelerinden oluşan bir gruptur.



Şekil 3.22 CNN Özellikleri ile Bölge Çıkarımı ve Bölgelerin Sınıflandırılması (Girshick vd. 2014)

Ne var ki, süreçte yer alan çoklu adımlar nedeniyle R-CNN oldukça yavaş hale gelir. R-CNN'in geliştirilmesiyle; Hızlı CNN (Fast CNN), Daha Hızlı CNN (Faster CNN), Maske (Mask) CNN gibi daha az maliyetli ve daha hızlı evrişimli sinir ağı modelleri kullanılmaya başlanmıştır.

3.5.1.2. Nokta Bulutu Temelli CNN Yöntemleri

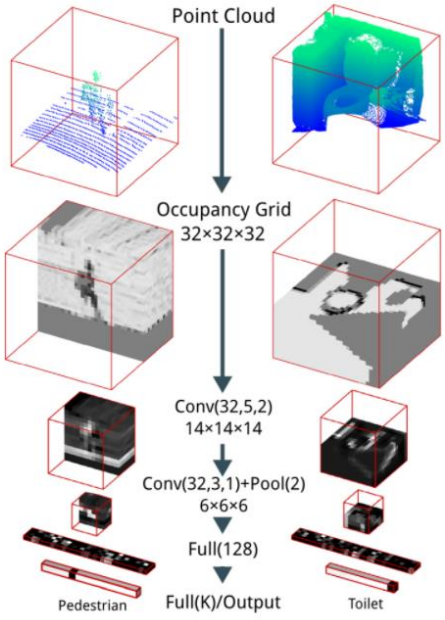
- **VoxNet**

Maturana ve Scherer (2015), temel olarak nokta bulutunu ve istenirse nokta bulutuna ait renk boyutu (RGB-D) ve yansıma değeri gibi ek özellikleri ve Bilgisayar Destekli Tasarım (Computer Aided Design – CAD) verilerini de kullanarak gerçek zamanlı obje tanıma yapabilen bir 3B CNN yaklaşımı geliştirmişlerdir. Arka planında dağınık başka nesneler de içerebilecek bir 3B nokta bulutu içerisinde, belirli bir nesnenin sınıf etiketini tahmin etme sorununu ele almışlardır.

Girdi nokta bulutu segmentinden oluşur. Segment genellikle nokta bulutu ile sınırlayıcı kutu (bounding box)'nun kesişimidir. Sistemleri ise Hacimsel Doluluk

Gridi (Volumetric Grid Occupancy) ve doğrudan bu gridden sınıf etiketini öngören bir 3B CNN olmak üzere iki ana bileşenden oluşur.

Doluluk gridi (occupancy grid), çevrenin durumunu, her biri bir voksele karşılık gelen rastgele değişkenlerin 3B bir kafes olarak temsil eder. Farklı bakış açılarından ve zaman örneklerinden gelen segmenti verimli bir şekilde tahmin etmeye yaramaktadırlar. Basit bir yapıları vardır, saklanabilirler ve değişiklik yapmaya elverişlidirler.



Hacimsel temsilde her nokta (x, y, z), ayrık voksel koordinatlarına (i, j, k) eşlenir. Eşleme tekdüze bir ayrıklaştırmadır, ancak uzaydaki voksel gridinin orijinine, yönüne ve çözünürlüğüne bağlıdır. Orijinin, girdi olarak verildiğini varsayılmaktadır.

Oryantasyon için, grid çerçevesinin z ekseninin yerçekimi yönüyle yaklaşık olarak hizalandığını varsayılmaktadır. Tüm deneylerde doluluk gridi boyutu sabit 32x32x32 voksel seçilmiştir.

Şekil 3.23 VoxNet

- **PointNet**

Girdi olarak nokta bulutunun modellerini ve RGB değerlerini kullanır. Çıktıda ise tüm girdinin sınıf etiketini veya her girdi parçası başına sınıf etiketini verir. PointNet'in temel fikri, her noktanın mekansal kodlamasını öğrenmek ve ardından tüm bireysel nokta özelliklerini global bir nokta bulutu imzasında toplamaktır.

İlk aşamada her nokta özdeş ve bağımsız olarak işlenir. Her nokta sadece x, y, z koordinatları ile temsil edilir. Normaller ve diğer yerel veya global özellikler hesaplanarak ek boyutlar elde edilebilir.

PointNet'in üç ana modülü vardır. Bunlardan birincisi, yaklaşımın anahtarı olan **simetrik fonksiyon** (maksimum havuzlama katmanı)'dur. Simetrik fonksiyon girdi olarak n vektör alır ve çıktı, girdi sıralamasından bağımsız bir vektördür. Örneğin, toplama ve çarpma operatörleri ikili fonksiyonlar için simetriktir. Vektörize edilmiş şekliyle, n tane noktadan f, g ve h fonksiyonları komşuluk özelliklerini oluştururlar.

$$f: 2^{\mathbb{R}^N} \to \mathbb{R}$$

$$h: \mathbb{R}^N \to \mathbb{R}^k$$

$$g: \mathbb{R}^k \times \ldots \times \mathbb{R}^k \text{ olmak üzere,}$$

$$f(\{x_1, \quad \ldots, x_n\}) \approx \underbrace{g(\{h(x_1), \quad \ldots, h(x_n)\})}_{\text{Simetrik fonksiyon}} \tag{3.11}$$

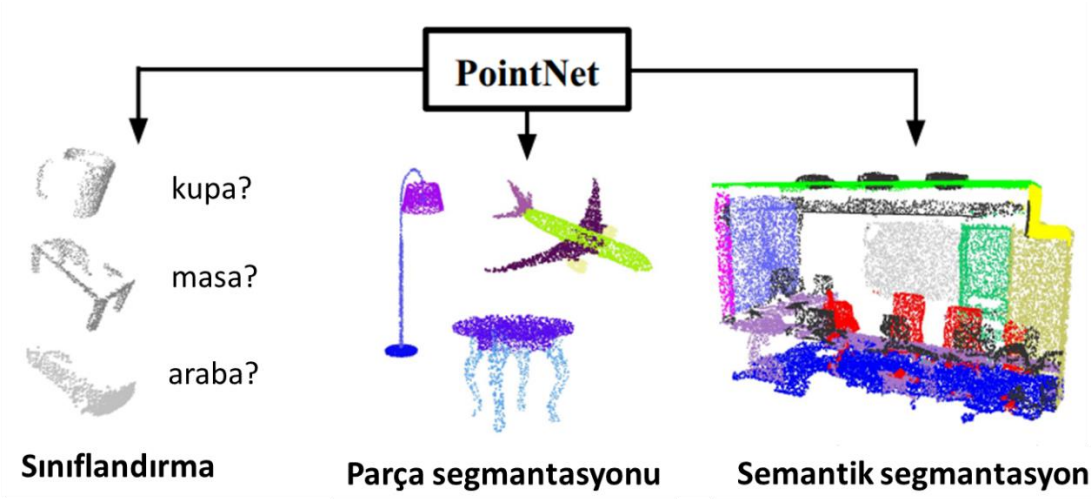
Burada h, girdi noktalarını özelliklerin çıkartılabileceği farklı bir temsile dönüştürürken, toplama işleminin simetri özelliği fonksiyonun girdi sırasından bağımsız olmasına izin verir. h, çok katmanlı bir algılayıcı ağ ile g, ise tek değişkenli bir fonksiyon ve maksimum havuzlama ile yaklaşık olarak tahmin edilir.

Ana modüllerden ikincisi ise **yerel ve global bilgi toplama** yapısıdır. Vektör formundaki çıktı, ona ait girdi verisinin global imzası olarak kabul edilmiştir. Nokta bulutunun global özellik vektörü hesaplandıktan sonra, global özellikle her noktanın özelliğini birleştirerek her noktaya geri besleme yapılmış ve her noktanın yerel bilgisi elde edilmiştir. Böylelikle ağa hem global hem de yerel geometriye dayalı özellikleri anlamlandırabilir tahmin edebilme yetisi kazandırılmıştır.

Ortak Uyum Ağı (Joint Alignment Network) ise üçüncü ana modüldür. T-Net olarak adlandırılan mini bir ağ eklenerek afin dönüşüm matrisleri belirlenmiş ve bu dönüşüm doğrudan gidi noktaların koordinatlarına uygulanmıştır. Bu mini ağın kendisi bile büyük bir ağa benzetilmiştir ve noktadan bağımsız özellik çıkarma, maksimum havuzlama ve tam bağımlı katmanlar (fully connected layer)'a sahiptir.

Ağ, etkili bir şekilde nokta bulutunun ilginç ve bilgi verebilecek seçimlerini ve seçim nedenlerini kodlayan bir dizi optimizasyon fonksiyonunu öğrenir. Ağın son tam bağımlı katmanı öğrenilmiş optimal değerleri tüm CAD modellerin sınıflandırması için toplar ve nokta etiketlerini tahmin etmek için kullanır.

PointNet'in uygulama alanları 3B obje sınıflandırma, 3B obje-parça segmantasyonu ve semantik segmantasyondur. 3B obje sınıflandırmada ağ, global nokta bulutu nesnelerini obje sınıflandırmasında kullanmak üzere öğrenir. 3B obje-parça segmantasyonunda amaç her noktaya veya yüzeye parça kategorisi etiketi (örn:koltuk ayağı, bardak tutacağı vs.) atamaktır. Semantik segmantasyonda ise karmaşık bir arka plan içerisindeki objeler etiketleriyle tespit edilir.



Şekil 3.24 PointNet'in uygulama alanları (Qi vd. 2017a)

PointNet'in sınıflandırma ağı, az önce bahsedilen simetrik fonksiyon, global ve yerel bilgi toplama bilgi kombinasyonu ve ortak uyum ağından meydana gelir. Bunların dışında bazı dönüşüm ağlarını ve seyreltme (dropout) katmanını da

içerir. İlk dönüşüm ağı (mini-PointNet), ham nokta bulutunu alır ve 3x3 boyutlu bir matrise dönüştürür. Mini-PointNet, bir paylaşımlı çok katmanlı algılayıcı (MLP), bir maksimum havuzlama katmanı ve iki tam bağımlı katmandan oluşur. Segmantasyon ağı ise sınıflandırmanın bir uzantısıdır.

PointNet'in segmantasyon ağında, yerel nokta özellikleri ve global özellikler her nokta için sıralanır. Bu ağda seyreltme katmanı yoktur. Eğitim paramatreleri sınıflandırma ağı ile aynıdır. Segmentasyon ağına nokta özelliği girdisi oluşturmak için, bu katmana farklı olarak girdinin sınıfını gösteren özel bir vektör eklenmiş ve bu maksimum havuzlama katmanının çıktısıyla birleştirilmiştir. Eklenen vektör, girdi şeklinin kategorisini belirtmek için kullanılmıştır. Ayrıca bazı katmanlardaki nöronlar arttırılmış ve farklı katmanlardaki yerel nokta özelliklerini toplamak için atlama bağlantıları eklenmiştir.

- **PointNet++**

Qi vd. (2017b), ilk versiyonun aksine komşuluk özelliklerini farklı ölçeklerde yani farklı arama çaplarında bir araya getirmişlerdir. Girdi nokta bulutunu merkezli ve ölçekli birim küreler halinde alt bölümlere ayırmışlardır.

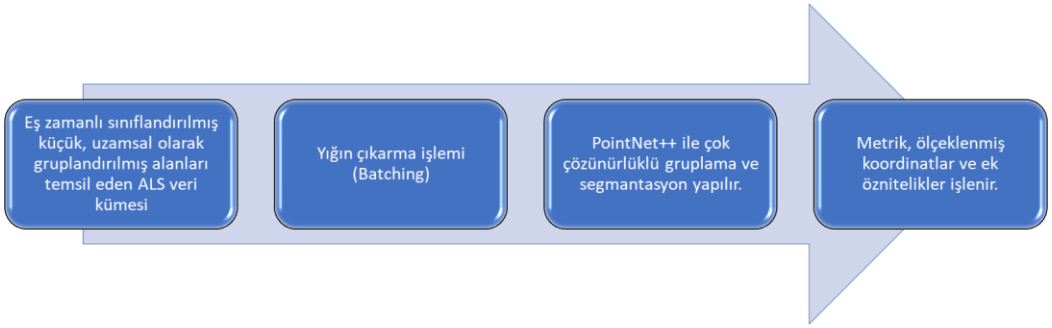
PointNet++, ilk noktaların gruplandığı ve özelliklerin toplandığı (kodlama) ve daha sonra özelliklerin sınıflandırılması için tek noktalara tekrar enterpolasyonun yapıldığı (kod çözme) bir kodlayıcı-kod çözücü ilişkisi kullanır.

Sınıflandırma görevi için, tüm komşuluk ölçeklerine göre oluşturduğu öznitelik vektörleri birleştirilir ve tam bağımlı katman yardımıyla birbirine bağlanır. Daha sonra Softmax fonksiyonu yardımıyla sınıflandırma gerçekleştirilir.

- **ALSNet**

ALSNet (Winiwarter vd., 2019), PointNet'in optimize edilmiş ve birden fazla ölçekte kullanılmasıyla elde edilen bir yöntemdir. PointNet'in odak noktası nokta bulutu modellerinden örneklenen ve RGB değerlerini içeren nesnelerin sınıflandırılmasıdır. Girdisi h fonksiyonu ile x, y, z koordinatları küreye ölçeklidir. ALSNet yönteminde ise girdi için koordinatların farkı komşuluk ağırılık merkezine göre kullanılmış; ancak ölçeklenmemiştir. Girdi olarak

noktaların mekansal dağılımı, sinyal gücü, çoklu yansımadaki ek bilgiler kullanılmıştır. Çıktıda her bir nokta bir sınıfa ayrı ayrı atanmıştır.



Şekil 3.25 ALSNet çalışma şekli

ALSNet yönteminde, ağın eğitiminde **yığın (batch)** adı verilen ve komşuluk bilgilerini içeren mini nokta yığınları kullanılmıştır. Her bir yığın ağdan eş zamanlı geçirilmiş ve eğitim sırasında ağırlık güncellemeleri yapılmıştır.

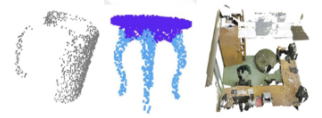
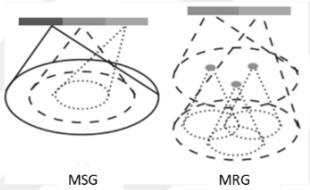
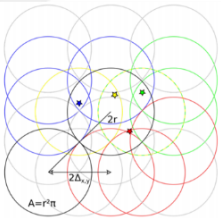
Yığınları oluşturmak için, bölgede grid oluşturarak düzenli aralıklı örneklem merkezleri seçilmiştir. Bu merkezlerin her birine en yakın k nokta seçilerek, bu k sayıdaki noktayı içeren yığın elde edilmiştir. Her noktanın birden çok yığın içerisinde kalması için yığınlar arasında kesişim alanları oluşturulmuştur. Böylelikle hesaplamadaki verimlilik arttırılmış, sınıflandırmada çoğulculuk sağlanmış ve komşuluk ilişkilerinin eksik kaldığı kenar alanlarda noktanın sınıfa atanma ihtimali arttırılmıştır. Yığın örnekleme mesafesi (daire yarıçapı) $\Delta x, y$ olmak üzere aşağıdaki eşlik ile hesaplanmıştır:

$$\Delta_{x,y} = \sqrt{\frac{k}{\pi\rho}} \cdot \frac{\sqrt{2}}{2} \cdot 0{,}95 \tag{3.12}$$

Bu eşlikte ρ ortalama nokta yoğunluğunu ve k yığının boyutudur ve bellek kısıtlamaları nedeniyle 200.000 seçilmiştir. 0.95 faktörü, tek tip olmayan nokta yoğunlukları için bir güvenlik marjı sağlamıştır ve veri setindeki maksimum ve minimum yerel nokta yoğunluklarına göre ihtiyatlı bir şekilde seçilmiştir.

Örtüşen yığınlar ile yapılan fazladan sınıflandırma, nihai sınıflandırmada karar verilirken kullanılmıştır. Her nokta için, ayrı sınıf olasılıklarının mevcut olduğu tüm yığınlar üzerinden ortalama alınmış ve bu ortalamanın argmax değeri en olası sınıf olarak değerlendirilmiştir. Benzer şekilde, sınıf olasılığına göre standart sapma her nokta için hesaplanabilmekte ve bir kalite ölçütü olarak kullanılabilmektedir.

Tablo 3.1 PointNet, PointNet++ ve ALSNet için girdi verileri hazırlarken kullanılan yığın yöntemlerinin kıyaslanması

• PointNet	• PointNet++	ALSNet
Girdi, direkt nokta bulutundan modellenmiş nesneler, nesne kısımları veya bir sahne içerisindeki nesne olabilir. 	Nokta bulutu, Öklid uzayında yerel özellikli alt bölgelere bölünür. Her bölüm içerisindeki noktalar, merkezi ve ölçeği olan bir komşuluk topu olarak tanımlanır. Girdi, bu komşuluk toplarından oluşur. Komşuluk küreleri içindeki nokta yoğunluğunu ayarlayabilmek için Multi çözünürlüklü gruplama (MRG), ölçeği ayarlayabilmek için ise Multi ölçekli gruplama (MSG) yöntemleri kullanılmıştır. 	Girdi, eşit aralıklı merkezlerin her birine en yakın k nokta seçilerek elde edilen nokta yığınlarıdır. 

4
ÇALIŞMA ALANI VE VERİ SETİ ÖZELLİKLERİ

Harita Genel Müdürlüğü'nden edinilen İzmir Bergama Lidar verisi üzerinde çalışılmıştır. Bu çalışmada iki farklı kırsal yerleşim alanı seçilmiştir. Çalışma alanları içerisinde bitki örtüsünün yanı sıra pek çok bina bulunmaktadır.

Ekim 2014'te gerçekleştirilen taramada Riegl LMS-Q1560 cihazı kullanılmıştır. Uçuş yüksekliği 1200m, uçuş hızı 150 knots, tarama açısı 35° (+/-), kolon sayısı 32, bindirme oranı %25 ve yoğunluk ≥ 8 nokta/m^2'dir (Kayı, Erdoğan, & Eker, 2015). Çalışmanın amacı nokta bulutu içerisindeki noktaları zemin, bina, bitki örtüsü olarak sınıflandırmak olduğu için ayrık dönüşlü Lidar verisi kullanmak yeterli görülmüştür.

Çalışma alanı koordinat sistemi WGS84 datumundadır, projeksiyonu ise Universal Transverse Mercator, Zone: 35'tir.

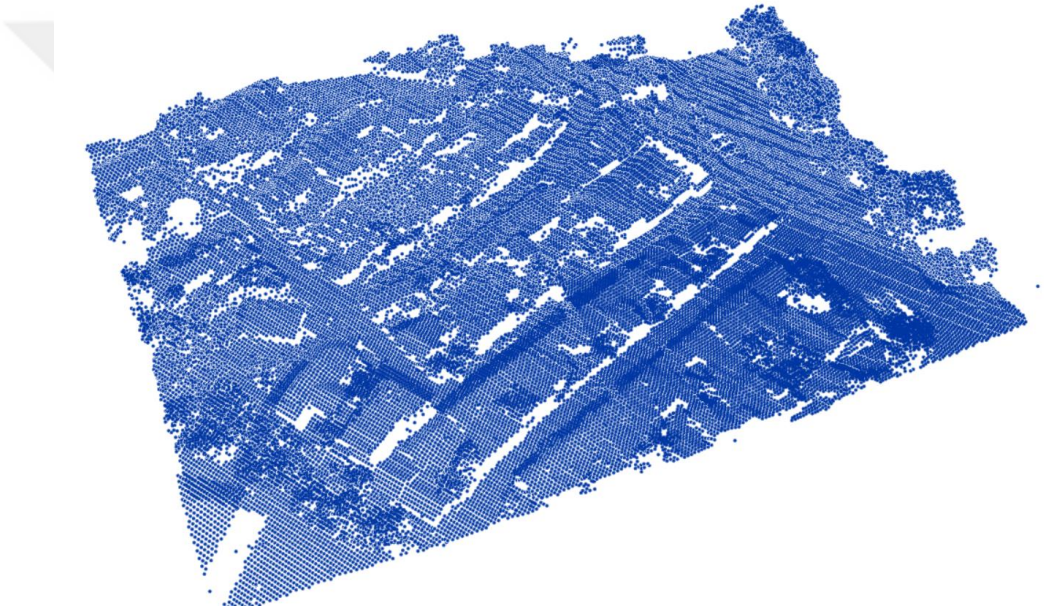
Birinci çalışma alanı eğimli bir alandır. Binalar tepenin yamacında sıralanmıştır. Ağaçlık alanlar toplu bir şekilde binaların dışındadır; ancak binaların arasında da tekli ağaçlar mevcuttur. İç içe yapılar arasında çatılardan uzanan saçaklar, farklı yükseklikteki yapı ve bahçe duvarları çalışma alanını karmaşık bir hale getirmektedir. Çalışma alanı 62 m x 120 m genişliğindedir. Çalışma alanı nokta bulutunda 48438 adet nokta mevcuttur.



Şekil 4.1 Birinci çalışma alanının 2013 yılına ait uydu görüntüsü



Şekil 4.2 Birinci çalışma alanının 2015 yılına ait sokak görüntüsü (tepenin yamacındaki binalar)



Şekil 4.3 Birinci çalışma alanına ait Lidar verisi

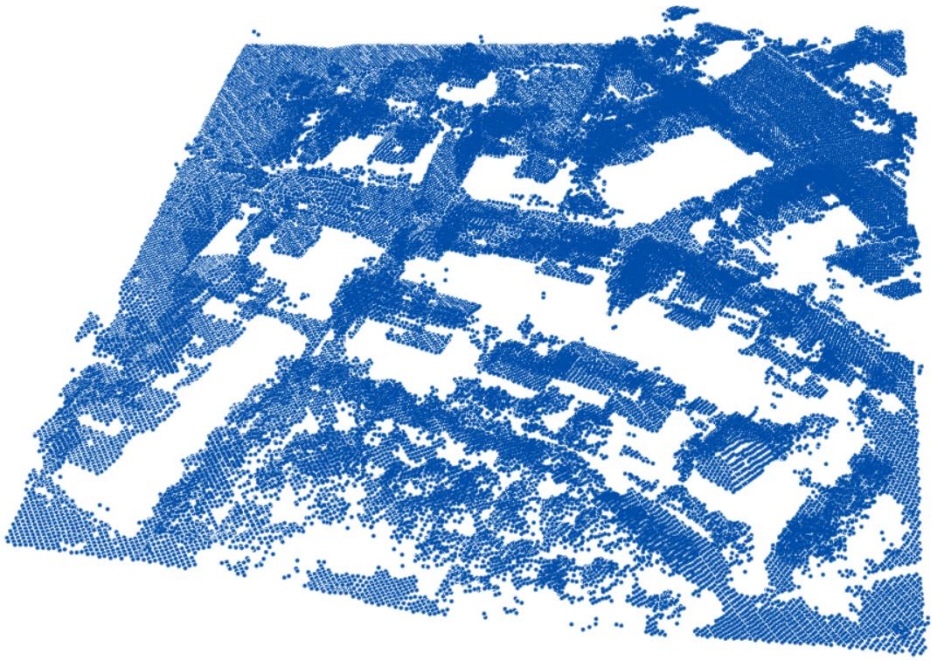
İkinci çalışma alanın ise birinci çalışma alanına göre eğimi ve kırsal özelliği daha azdır. Ağaçlık alanlar görselin alt kısmında toplanmış olsa da binaların arasında da tekli ağaçlara rastlanmaktadır. Çalışma alanı nokta bulutunda 58725 adet nokta mevcuttur.



Şekil 4.4 İkinci çalışma alanının 2013 yılına ait uydu görüntüsü



Şekil 4.5 İkinci çalışma alanı içerisinden bir sokağın 2015 yılına ait sokak görüntüsü



Şekil 4.6 İkinci çalışma alanına ait Lidar verisi

5
METODOLOJİ

5.1 Kendini Düzenleyen Haritalar

KDH ile sınıflandırmada, ayrık-dönüşlü Lidar noktaları, nokta tabanlı bir yaklaşımla sınıflara ayrılmıştır. KDH ağının giriş parametreleri, noktaya ait *x-y-z* koordinatları ve yansıma (intensity) bilgisidir. Yansıma bilgisi olarak üretilen raster görüntünün gri değerleri kullanılmıştır. Değişkenlere ait ortalama ve varyans değerleri birbirinden anlamlı derecede farklı olduğunda değişkenlerin 0-1 aralığına normalize edilmesi KDH'nin performansını arttırmaktadır. Giriş değişkenleri Eşitlik 5.1 ile verilen, minimum-maksimum yöntemi ile normalize edildikten sonra KDH ile kümeleme işlemine geçilmiştir.

$$X_i' = (X_i - X_{min})/(X_{max} - X_{min}) \tag{5.1}$$

Burada; X_i', normalizasyon değerini; X_i, öznitelik değerini; X_{min}, en küçük öznitelik değerini ve X_{max}, en büyük öznitelik değerini ifade etmektedir.

Normalize edilen veriler kullanılarak denetimsiz bir yapay sinir ağı yöntemi olan KDH yöntemiyle hava Lidar noktalarının sınıflandırılması aşağıdaki adımlar kullanılarak sağlanmıştır. İş akış diyagramı Şekil 5.1'de verilmiştir.

i) Sistematik bir şekilde az sayıda küme oluşturularak en etkili nokta sınıflandırılmasının belirlenmesi amaçlanmıştır. Bu nedenle, başlangıç nöron sayısı iki olarak belirlenmiştir. Bu bağlamda, küme sayısını veren ağın boyutuna kümeleme performansı göz önünde bulundurularak karar verilmiştir.

ii) Kümelere ayrılan noktalar, uydu görüntüleri ile görsel olarak karşılaştırılarak etiketlendirilmiştir. Bu aşamada aynı nesne sınıfına ait nokta kümeleri birleştirilerek nokta sınıfları belirlenmiştir.

iii) Tüm noktaların görsel olarak sınıflandırıldığı test verisi ile KDH yapay sinir ağı kullanılarak belirlenen nokta sınıfları karşılaştırılarak, zemine ait noktaların ve zemine ait olmayan noktaların sınıflandırılması, bir başka deyişle filtrelenmesi ile ilgili Tip-1, Tip-2 ve toplam hataları belirlenmiştir (Sithole & Vosselman, 2003).

$$Tip\ 1\ hatası = b/(a + b) \tag{5.2}$$

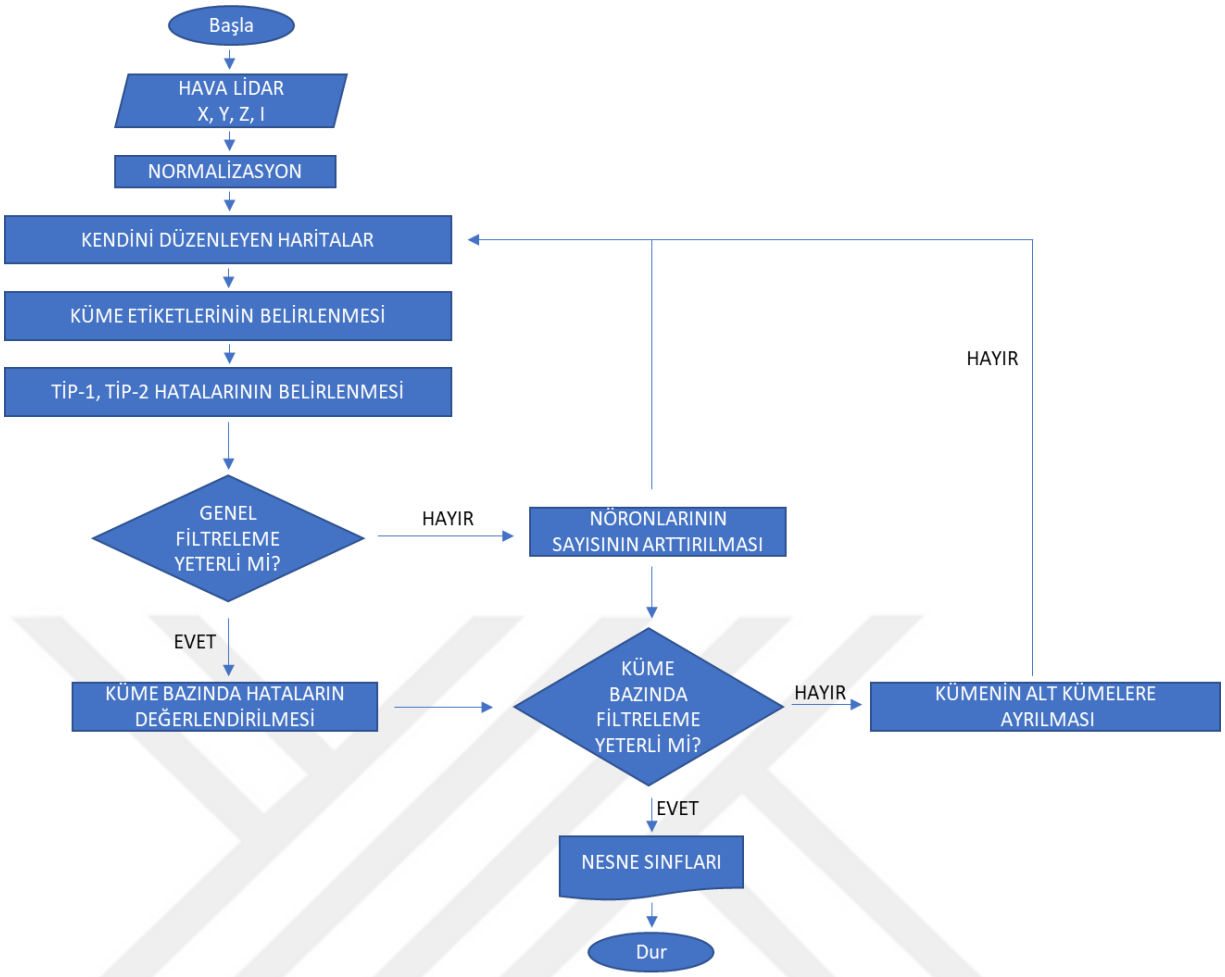
$$Tip\ 2\ hatası = c/(c + d) \tag{5.3}$$

$$Toplam\ hata = (b + c)/e \tag{5.4}$$

Burada; a, doğru sınıflandırılmış zemine ait noktaları; b, zemine ait olmayan nokta olarak yanlış sınıflandırılmış zemine ait noktaları; c, zemine ait nokta olarak yanlış sınıflandırılmış zemine ait olmayan noktaları; d, doğru sınıflandırılmış zemine ait olmayan noktaları; e, toplam test nokta sayısını göstermektedir.

iv) Noktaların tümünde genel anlamda filtrelemenin yeterli olması için toplam filtreleme hatasının tolerans değerini geçmemesi gerekmektedir. Yeterli doğruluğa ulaşılamaması durumunda ağ topolojisindeki nöron sayısı sırasıyla x ve y yönünde bir arttırılır.

v) Noktaların tümünde istenilen filtreleme doğruluğuna erişildikten sonra, küme bazında toplam filtreleme hatasının tolerans değerini geçmemesi gerekmektedir. Yeterli doğruluğa ulaşılamaması durumunda küme, alt kümelere ayrılmak amacıyla KDH yöntemi tekrar uygulanır.



Şekil 5.1 KDH iş akış diyagramı

5.2 Rastgele Orman Algoritması

Hava Lidar nokta bulutu eğitim ve test verisi olmak üzere ayrılarak direkt algoritma girdisi olarak kullanılmıştır.

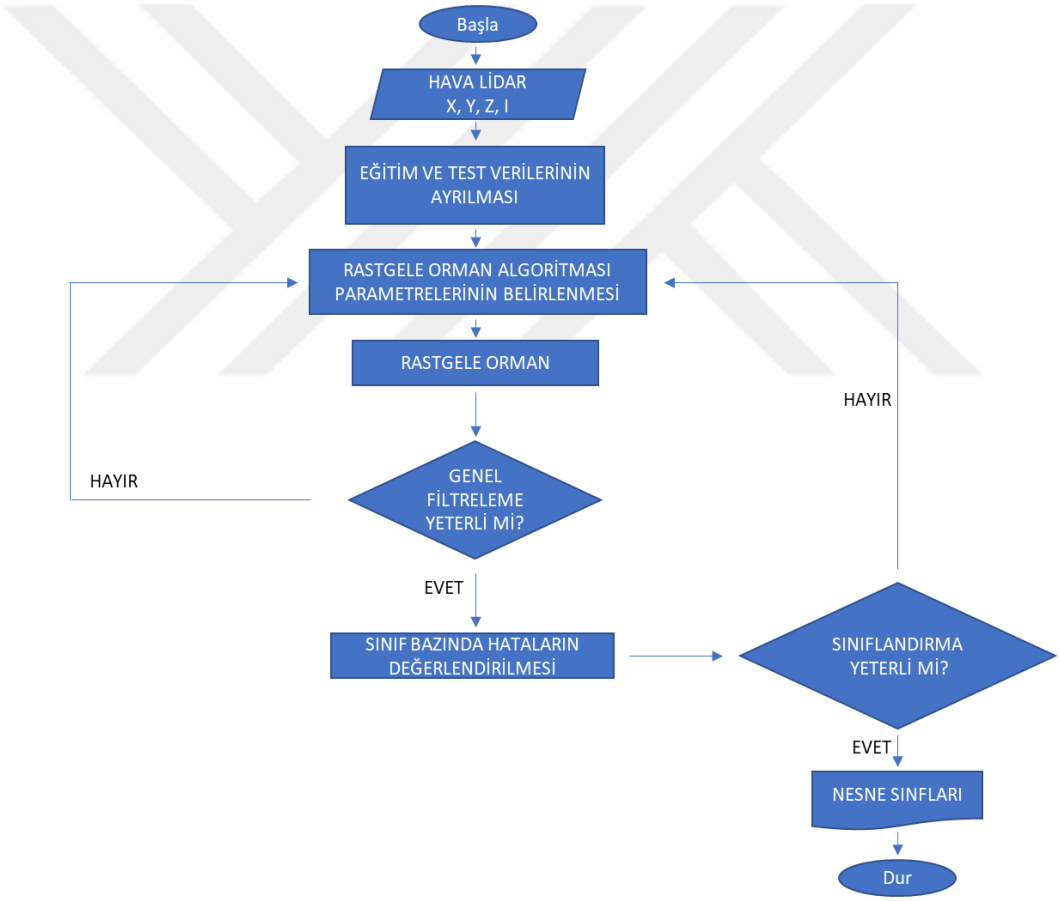
Rastgele Orman Algoritması oldukça sade, anlaşılır ve kodlaması pratik bir yöntemdir. Bu çalışmada eğitim için kullanılan parametreler ağaç sayısı, ağacın maksimum derinliği ve rastgelelik faktörü kullanılmıştır. Şekil 5.2'deki iş akış diyagramında görüldüğü üzere bu parametreler deneysel olarak belirlenmiştir.

Öncelikle noktaların zemin ve zemin olmayan olarak ne kadar başarılı filtrelendiği 2x2 boyutlu bir hata (confusion) matrisi ile kontrol edilmiştir. Filtreleme doğruluğu yeterli görüldüğü takdirde bir sonraki aşama olan sınıflandırmaya geçilmiştir.

Doğruluk yeterli görülmediği takdirde parametrelerin yeniden belirlenmesi adımına geri dönüş söz konusu olmuştur.

Sınıflandırma aşamasında noktaların zemin noktaları, bitki noktaları ve bina noktası olmak üzere 3 nesne grubuna ayrılması hedeflenmiştir. Sınıflandırma doğruluğu 3x3 boyutlu bir hata matrisi ile yeniden kontrol edilmiştir. Sınıflandırma doğruluğu yeterli görüldüğü takdirde sınıflandırma sonuçları yazdırılmıştır, sonuçların yeterli olmadığı takdirde yine parametrelerin güncellenmesi yönünde geriye dönüş olmuştur.

Bu yöntemde nokta bulutuna normalizasyon uygulamadan ve ağırlık kullanılmadan dahi iyi bir sınıflandırma sonucu elde edilmiştir.



Şekil 5.2 RO algoritmasının iş akış diyagramı

5.3 Derin Öğrenme

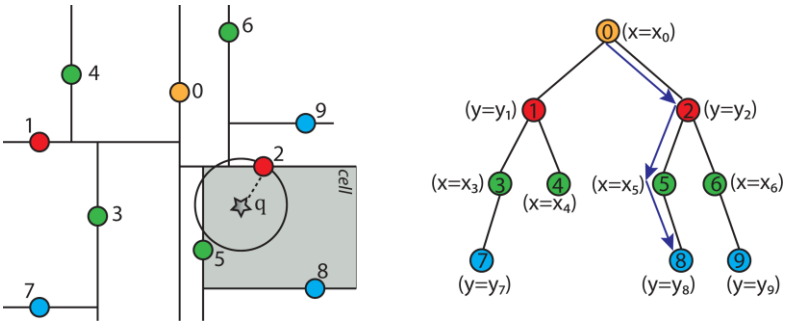
Nokta bulutu eğitim ve test verisi olmak üzere ikiye ayrılmıştır. Verinin *x* ve *y* koordinat değerlerine minimum-maksimum normalizasyonu uygulanırken, *z* ve *i* değerlerine uygulanmamıştır. Böylelikle yükseklik ve yansıma değerlerinin derin öğrenmede daha baskın bir etkiye sahip olması hedeflenmiştir ve özellikle bir ağırlık belirtmeye gerek kalmamıştır.

Birbiriyle kesişimli yığınları oluşturmak için öncelikle bir grid yardımıyla yığın merkezleri ve aralarındaki mesafeler belirlenmiştir. Yığın merkezleri de eğitim ve test noktalarına uygun düşecek şekilde ayrılmıştır. Bu aşamada eğitim verisi noktalarının hiçbirinin test yığınlarına, test verisi noktalarının hiçbirinin eğitim yığınlarına düşmemesi önemli ve gereklidir. Ayrıca yığın oluştururken, her noktanın en az bir yığın içinde kalması ve mümkünse birden fazla yığın içerisinde kalarak birden fazla sınıflandırma değerinin olması, daha sonra bu sınıflandırma değerlerinden çoğulculuk ilkesine göre en çok tekrar eden sınıf değerinin seçilmesi esas alınmıştır. Bu nedenle mümkün olduğunca sık ve yarıçaplarıyla örtüşebilen merkezler alınmıştır.

Seçilen eğitim ve test yığın merkezleri, kendi veri setlerinin içinde *k*-boyutlu ağaçlar (k-DTree) algoritması ile kendine en yakın komşu noktaları seçerek yığınları oluşturmuştur. Her iki test alanı için de komşu nokta sayısı 300 olarak uygun görülmüştür. Yani her bir yığın içerisinde 300 nokta bulunmaktadır. Ancak merkezler arası mesafe ve yarıçap iki alan için farklı seçilmiştir.

K-boyutlu ağaçlar, en yakın komşuluk analizinde kullanılan hızlı bir ikili ağaç yöntemdir. K-Boyutlu ağaçların her bir düğümünün ekseni ile boşlukları hiperdörtgenlere bölünmektedir. Şekil 5.3'te görüldüğü gibi köklerden (root) yapraklara (leaf) gidildikçe uzay, her bir düğüm *x* ve *y* eksenlerince iteratif olarak ikiye bölünür.

Yapılandırma sırasında, eksen ve bölme noktası hücrelerinin hepsinin uzun ve ince olmamasını sağlayan "kayan orta nokta kuralı" kullanılır. Herhangi bir noktanın *r* yarıçapındaki en yakın komşuları için sorgulama yapmak verimi büyük ölçüde arttırabilir.



Şekil 5.3 K-boyutlu ağaçların çalışma şekli (Lixin vd, 2014)

K-boyutlu ağaçların çalışma şekli:

1. Şekilde her bir düğüm bir dikdörtgeni temsil eder.
2. Önce *x* eksenine göre ortadaki nokta seçilerek noktalar sağ ve sol olmak üzere ikiye ayrılır.
3. *y* eksenine göre ayrılan kısımlarda ortadaki noktalar seçilir ve orta noktaların *x* eksenine göre tekrar ikiye ayrılır.
4. Bu işlemler iteratif olarak hem sağ hem de sol tarafta tekrarlanır. Sağdaki değerler sağa, soldaki değerler sola yazılır.

Arama noktasına en yakın komşuyu bulmak için,

1. Sağa mı, sola mı dönüleceğini tespit etmek için arama noktasının ilk seçilen noktadan büyük mü, yoksa küçük mü olduğuna bakılır.
2. Eğer arama noktası dikdörtgenin içindeyse, dikdörtgenin içindeki her bir noktaya mesafesi hesaplanır ve en yakın nokta bulunur. Aradaki mesafeden *r* yarıçaplı çember geçirilir ve çemberin içerisinde başka nokta olup olmadığı kontrol edilir.
3. Eğer çember içine giren başka nokta varsa başka dikdörtgene geçilir. Çemberin içerisine nokta girmeyene kadar iteratif olarak ilerlenir.

Yığınlar oluşturulurken aynı zamanda veri seti noktaları indekslenmiştir. Böylelikle çıktı olarak sınıf sonuçları alındığında her noktanın kaç yığın içine girdiği, kaç sınıflandırma değeri aldığı ve bu sınıflandırma değerlerindeki farklılığın ne kadar olduğu gibi pek çok bilgiye erişim de sağlanmıştır.

Sonraki aşamada eğitim verisinin zenginleştirilmesi yapılmıştır. Bunun için PointNet modelinde kullanılan Dönme (Rotate) ve Titreşim (Jitter) yöntemleri

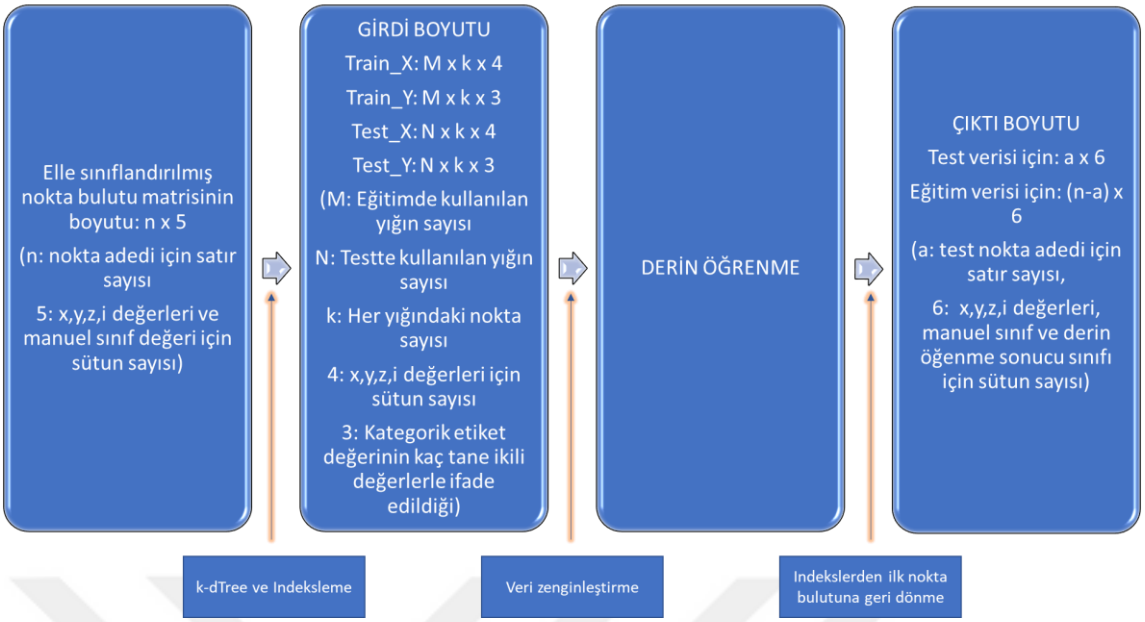
seçilerek eğitim verisinin farklı varyasyonları elde edilmiştir. Ancak bu adıma yansıma değerlerinin sabit kalması gerektiği düşünülerek yansıma değerleri dahil edilmemiştir. Dönme yöntemi ile her bir yığın içerisindeki noktalar, dönüklük matrisi kullanılarak döndürülmüştür. Kullanılan dönüklük matrisinin en önemli özelliği yükseklik değerini değiştirmemesi, yalnızda *x* ve y koordinat değerlerini değiştirmesidir. Şekil 5.4'te gösterilen dönüklük matrisinde θ vektörün dönüş yönünü temsil etmektedir.

cosθ	-sinθ	0
sinθ	cosθ	0
0	0	1

Şekil 5.4 Dönüklük matrisi

Titreşim ise, yığın içindeki noktaların hafifçe yerini değiştirmeyi sağlayan bir yöntemdir. Bu yöntem uygulanırken, *x* ve *y* değerleriyle birlikte *z* değeri de çok az değiştirilmiştir.

Etiket girdisi verisi, her bir sınıf ikili olarak üç değer ile kategorik olarak kodlanmıştır. Model girdisi için hazırlanan diziler Train_X, Train_Y, Test_X ve Test_Y olarak kaydedilmiştir. Train_X, her biri 300 noktadan oluşan *x, y, z, i* değerlerini içeren eğitim yığınlarının oluşturduğu dizidir. Train_Y ise her eğitim yığını içerisindeki her noktanın eğitim için gerçek etiket değerini içerir. Test_X, test edilmek üzere her biri 300 noktadan oluşan *x, y, z, i* değerlerini içeren yığınların oluşturduğu dizidir. Son olarak, Test_Y ise her test yığını içerisindeki her noktanın eğitim için gerçek etiket değerini içerir. Test_X ve Test_Y eğitim aşamasında kullanılmaz. Test_Y değeri yanlızca tahmin sonucunun doğruluğunu ölçebilmek için kullanılır.



Şekil 5.5 Derin öğrenmede kullanılan girdi versi ve çıktı veri

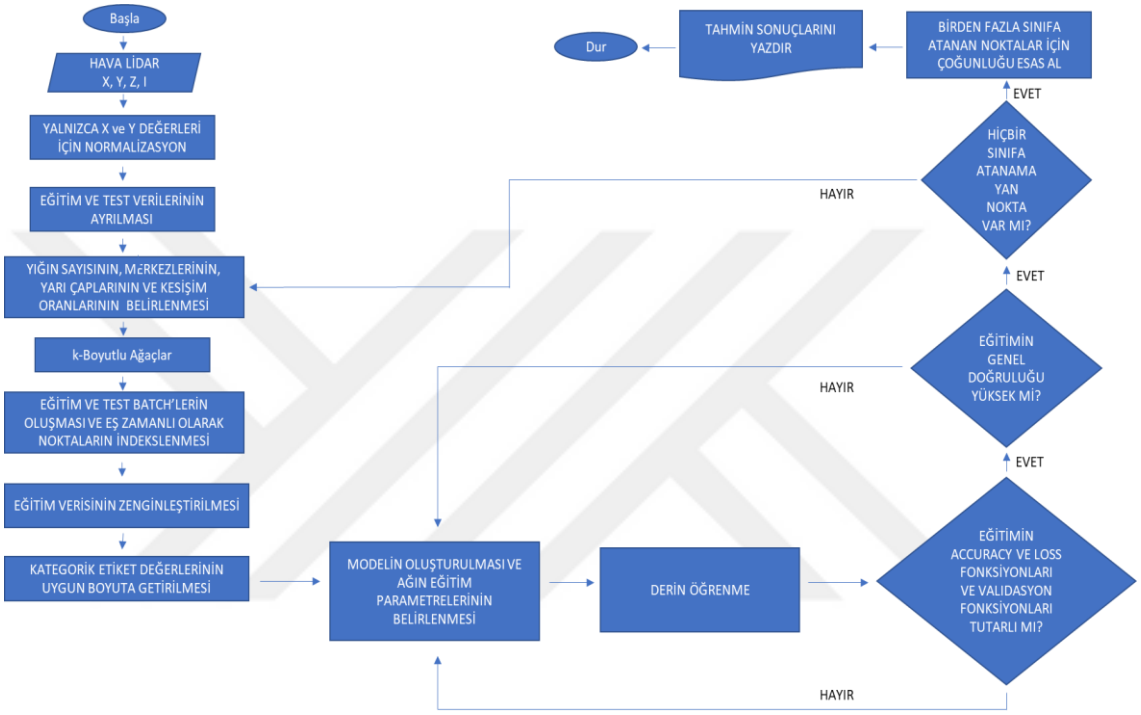
Ağ modeli oluşturulurken PointNet'in segmantasyon adımındaki modelin bir parçası kullanılmıştır. Model evrişimli sinir ağlarını, yığın normalizasyonunu, simetri fonksiyonunu, maksimum havuzlama katmanını içermektedir. Epok değeri 100, optimizasyon için Adam optimizasyon aracı ve öğrenme oranı 0.01 seçilmiştir. Son katmanında Softmax, diğer katmanlarında RelU aktivasyon fonksiyonları kullanılmıştır. Modelde PointNet'in segmantasyon adımında olduğu gibi seyreltme katmanı bulunmamaktadır.

Eğitim sonrasında, kesinlik (precision), duyarlılık (recall), f1-skor değerleri hesaplanmıştır. Sınıflandırma doğruluğu (accuracy) ve kayıp fonksiyon değeri kontrol edilmiştir. Doğruluk ve kayıp fonksiyonlarının grafiklerine bakılarak tutarlılıkları değerlendirilmiştir. Sınıflandırma doğruluğu yeterli değilse veya doğruluk ve kayıp fonksiyonu ile tutarlı bir eğilim göstermiyorsa model yeniden kontrol edilmiştir.

Eğer sınıflandırmadan sonra test noktalarından hiçbir sınıfa atanamamış veya hiçbir yığın içerisine girememiş nokta varsa, k-boyutlu ağaçlar ile yığınlama aşamasına geri dönülerek yığın merkezleri sıklaştırılıp, yığın yarıçapları arttırılarak daha

büyük kesişim alanı elde edilmiştir. Böylelikle tüm test noktalarının en az bir yığın içerisine alınması sağlanmıştır.

Son olarak, sınıflandırma değerleri indekslemeye bağlı kalınarak yazdırılır, etiketler ikili değerlerden tekrar kategorik veriye dönüştürülür ve çıktı olarak sınıflandırılmış test ve eğitim nokta bulutları alınır.



Şekil 5.6 Derin öğrenme iş akış diyagramı

<u>Doğruluk, kesinlik, duyarlılık ve f1-skor değerlerinin hesaplanması:</u>

Sınıflandırma sonrası elde edilen hata matrisi (confusion matrix) üzerinde tahminlenen ve gerçekleşen değerler arasındaki ilişki gösterilir. Başka bir deyişle, noktaların kaçının doğru sınıfa atandığı, kaçının yanlış sınıflara atandığı tespit edilir.

Doğru pozitif ve Doğru negatif değerleri modelin doğru tahminlediği, Yanlış pozitif ve Yanlış negatif değerleri ise modelin yanlış tahminlediği noktaları göstermektedir.

	Tahminlenen (Predicted)	
Gerçekleşen (Actual)	Doğru Pozitifler (DP)	Yanlış Negatifler(YN)
	Yanlış Pozitifler (YP)	Doğru Negatifler(DN)

Şekil 5.7 Hata matrisi örneği

Matrisin elemanlarını örnek ile açıklamak gerekirse,

- Doğru pozitif: Zemine ait nokta olarak filtrelenen noktanın aslında da zemine ait nokta olması.
- Doğru negatif: Zemine ait olmayan nokta olarak filtrelenen noktanın aslında da zemine ait nokta olmaması.
- Yanlış pozitif (Tip 1 Hatası): Zemine ait olmayan nokta olarak filtrelenen noktanın aslında zemine ait olması.
- Yanlış negatif (Tip 2 Hatası): Zemine ait nokta olarak filtrelenen noktanın aslında zemine ait olmayan nokta olması.

Hata matrisinin elemanlarından aşağıdaki eşitlikler yardımıyla doğruluk, kesinlik, duyarlılık ve f1-skor değerleri hesaplanır (Liu & Lang, 2019):

$$Doğruluk = \frac{DP + \mathrm{D}N}{DP + YP + \mathrm{Y}N + \mathrm{D}N} \tag{5.5}$$

$$Kesinlik = \frac{DP}{DP + YP} \tag{5.6}$$

$$Duyarlılık = \frac{DP}{DP + YN} \tag{5.7}$$

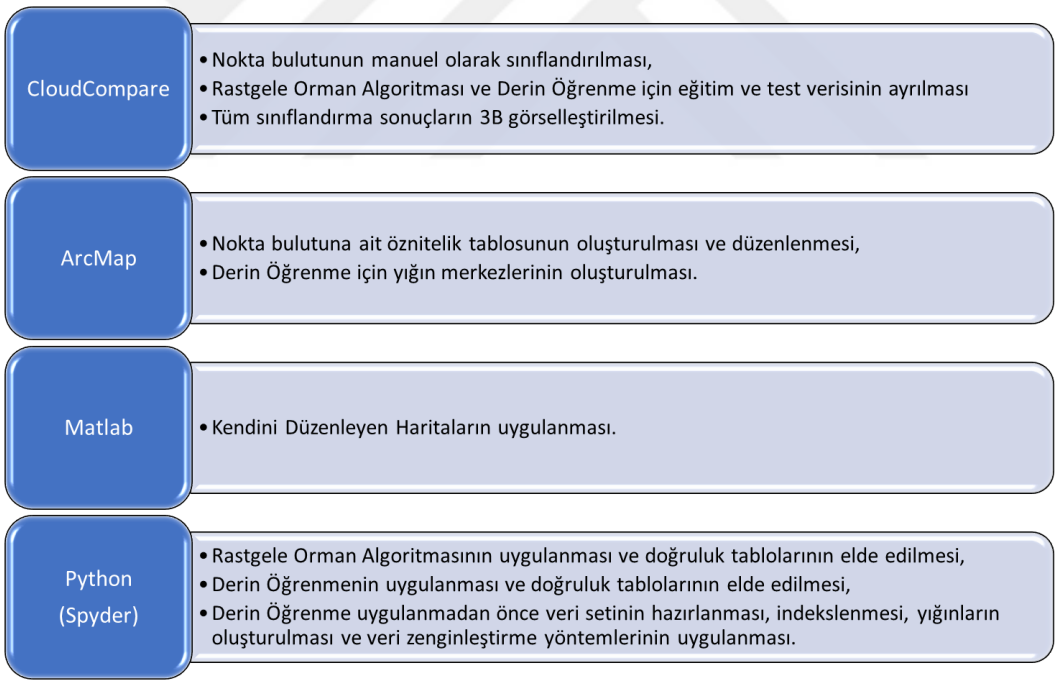
$$F1 - skor = \frac{2\ x\ Kesinlik\ x\ Duyarlılık}{Kesinlik\ +\ Duyarlılık} \tag{5.8}$$

6
KULLANILAN ARAÇLAR VE PARAMETRELER

Uygulama aşamasında kullanılan yazılımlar ve programlama dilleri Şekil 6.1'de gösterilmiştir.

İlk aşamada, nokta bulutu verisi ArcMap ortamına aktarılarak las verisinden her bir nokta için ayrı bir satırı olan ve her satır için *x, y, z* koordinatları ve *i* yansıma bilgileri birer sütunda olan nokta bulutu elde edilmiştir.

Öznitelik tablosu olan noktalar, sınıflandırma doğruluğunu ölçebilmek için manuel olarak hassas bir şekilde sınıflandırılmıştır ve sınıfları ayrı bir sütuna kaydedilmiştir. Bu aşamadan sonra nokta bulutu üç ayrı yapay zeka yönteminde girdi olarak kullanılmaya hazır hale gelmiştir.

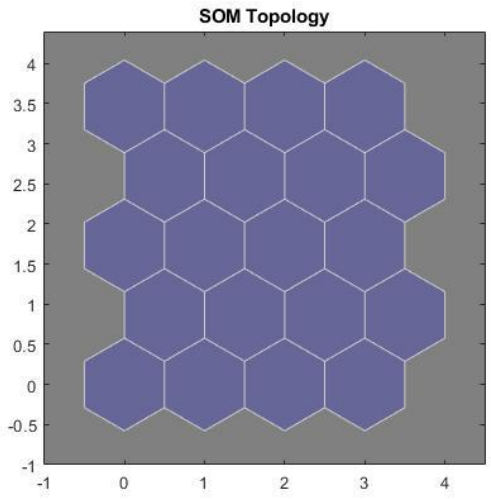


Şekil 6.1 Kullanılan yazılımlar ve programlama dilleri

6.1 Kendini Düzenleyen Haritaların Uygulanması

Kendini Düzenleyen Haritalar, denetimsiz bir yapay sinir ağı yöntemi olduğu için ağın eğitimi aşamasında etiket değerlerine ihtiyaç yoktur. Bu yüzden girdi için oluşturulan öznitelik tablosundan etiketlerin bulunduğu sütun silinmiştir.

Nokta sayısı *n* ve *x, y, z, i* sütunlar olmak üzere *nx4* boyutlu matris Matlab yazılımına girdi veri olarak aktarılmıştır. KDH ile filtreleme ve sınıflandırma için Matlab'in KDH yapay sinir ağları araçları kullanılmıştır. Her iki çalışma alanında da altıgen topoloji seçilmiş ve 200 iterasyon yapılmıştır.



Şekil 6.2 KDH topolojisi

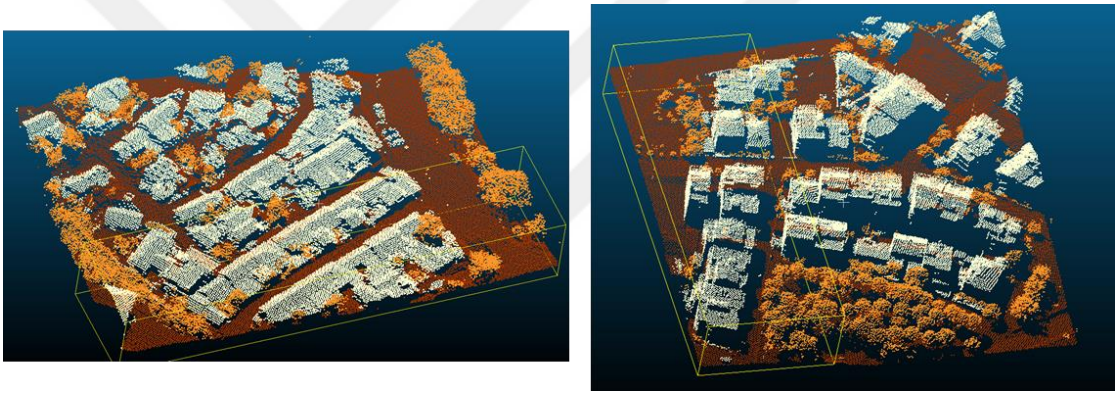
Her iki çalışma alanında da yükseklik ve yansıma değerlerinin ağa giriş ağırlığı *x* ve *y* değerlerine göre daha çok alınmıştır Şekil 5.1'deki iş akış diyagramına göre ağın boyutu 4x5 olarak seçildiğinde filtreleme sonuçlarının en iyi olduğu görülmüştür ve toplamda 20 adet küme oluşturulmuştur. Oluşan 20 kümenin içerisinde nesne noktalarını kümelenme doğruluğuna bakılmıştır. Nesne kümeleme doğruluğu yeterli olan kümelere nesne etiketi atanmıştır. Zemin, bitki, bina noktalarının karıştığı yani nesne kümeleme doğruluğu yeterli olmayan kümelere kendi içerisinde bir kez daha KDH uygulanmıştır. Filtreleme ve kümeleme doğruluğu daha yüksek çıkan 3x1 topolojisine göre üçlü alt kümeler oluşturulmuştur ve üçlü alt kümelerin nesne etiketleri atanmıştır.

Nesne etiketleri atandıktan sonra KDH ile sınıflandırılan nokta bulutu ve manuel sınıflandırılan nokta bulutu 3B görüntülenmiştir. Öznitelik tabloları yardımıyla hata matrisi oluşturulmuş ve sınıflandırma doğruluğu ölçülmüştür.

6.2 Rastgele Orman Algoritmasının Uygulanması

Denetimli bir sınıflandırma yöntemi olduğu için sınıf etiketlerine ihtiyaç vardır. Nokta bulutunun etiketli öznitelik tablosu kullanılmıştır. Verinin eğitim ve test verisi olarak ikiye ayrılması CloudCompare yardımıyla gerçekleştirilmiştir.

Eğitim ve test veri setleri Şekil 6.3'te görüldüğü gibi ayrılmıştır (Test veri setleri sarı dikdörtgen prizma içerisindeki nokta bulutudur). Veri seti ayrılmadan önce yapılan manuel sınıflandırmada kahverengi noktalar zemin, turuncu noktalar bitki, beyaz noktalar ise bina sınıfını temsil etmektedir.



Şekil 6.3 Eğitim ve test veri setlerinin ayrılması

Birinci çalışma alanında eğitim veri setinin ortalama yüksekliği, test veri setine göre daha fazladır. Eğitim verisi %63.25, test verisi %36.75 oranındadır. İkinci çalışma alanında ise çok bariz bir yükselti farkı yoktur. Eğitim verisi %65.16, test verisi %36.75 oranındadır.

RO algoritmasının diğer algoritmalara göre daha sade ve uygulaması kolay olması, yığınlama ihtiyacını ortadan kaldırmıştır. Her bir nokta direkt olarak sınıflandırılmıştır.

İki çalışma alanı kırsal alan olsa da gerek arazi eğimi gerekse bina düzeni açısından iki alan birbirinden farklı özellikler gösterebilmektedir. İkinci çalışma alanı biraz

daha kentsel özellik taşımaktadır. Bu sebeple iki alan için de eğitim ve test verisinin ayrılma şekli farklı yapılmıştır.

RO algoritmasını uygulamak için Python kütüphanesinin Pandas, NumPy ve Scikit-Learn kütüphaneleri kullanılmıştır ve aşağıdaki temel ifade, duruma göre parametreler değiştirilerek ve opsiyonlar eklenerek çalıştırılır [URL4]:

*RandomForestClassifier(n_estimators=***, max_depth=***, random_state=***)*

n_estimators: Ormandaki ağaç sayısı belirtir.

max_depth: Ağacın maksimum derinliğini belirtir.

random_state: ("Bootstrap = True" ise) Hem ağaçları oluştururken yeniden örneklemenin rastgeleliğini, hem de her düğümde en iyi bölünmeyi ararken dikkate alınacak özelliklerin örneklemesini kontrol eder.

Bu çalışmada algoritma parametreleri,

n_estimators:120, max_depth:36 ve random_state:60 seçilmiştir.

Varsayılan olarak Bootstrap=True'dur.

Algoritma çalıştırılmış ve daha sonra oluşan hata matrisleri incelenmiştir. Ayrıca *x, y, z, i* değerlerinin filtreleme ve sınıflandırmayı ne kadar etkilediğini gösterir grafik çizdirilmiştir.

6.3 Derin Öğrenmenin Uygulanması

Rastgele orman algoritmasında olduğu gibi Derin Öğrenme için de eğitim sırasında sınıf etiketlerine ihtiyaç vardır. Derin öğrenmede girdi olarak, iki alan için de rastgele orman algoritmasında kullanılan eğitim ve test verilerinin aynısı kullanılmıştır. Böylelikle iki çalışma alanı için de hem derin öğrenme, hem de rastgele orman ile elde edilen sonuçları kıyaslayabilme imkânı bulunmuştur.

Eğitim ve test verilerine k-boyutlu ağaçlar yöntemiyle yığınlama yapılmıştır. Bunun için öncelikle veri setindeki en büyük ve en küçük *x, y* değerleri tespit edilmiştir. Bu değerlere göre eşit aralıklı yığın merkezleri oluşturulmuştur. Şekil 5.2'deki iş akış diyagramında görüldüğü gibi her nokta bir yığın içerisine yerleşip, test sonucunda en az bir sınıf etiketi alıncaya kadar yığın sıklığı arttırılmıştır.

Yığınlama (batching) aşamasında Python 3.6.9 programlama dili ve Spyder ara yüzü kullanılmıştır. NumPy kütüphanesiyle birlikte özel olarak veri bilimi problemlerini çözmeye yarayan SciPy kütüphanesi kullanılmıştır.

SciPy kütüphanesinin cKDTree sınıfı, herhangi bir noktanın en yakın komşularını hızlı bir şekilde aramak için kullanılabilen bir dizi k-boyutlu noktaya bir indeks sağlar. Aşağıda en temel ve opsiyonsuz ifadesiyle kullanılan query_ball_point ise, *x* noktalarının *r* (yarıçap) mesafesi içerisindeki komşu tüm noktalarını bulduran bir k-boyutlu araç yöntemidir [URL5].

cKDTree.query_ball_point(self, x, r)

Derin öğrenme uygulamasını gerçekleştirebilmek için Tensorflow kütüphanesinden, Keras modelinden ve katmanlarından yararlanılmıştır. Doğruluk analizleri içinse Scikit-Learn araçları kullanılmıştır.

Birinci çalışma alanında, eğitim verisine ait 1020 ve test verisine ait 612 yığın olmak üzere, toplam yığın sayısı 1623'dir. Son duruma göre yığın merkezleri arasındaki mesafe 2.6 m ve yarıçapları 5.2 m olarak seçilmiştir. Böylelikle her yığın arasında %50 oranında kesişim sağlanmıştır.

İkinci çalışma alanında ise, eğitim verisine ait 3600 ve test verisine ait 2100 yığın olmak üzere, toplam yığın sayısı 5700'dür. Son duruma göre yığın merkezleri arasındaki mesafe 2 m ve yarıçapları 6 m olarak seçilmiştir. Böylelikle her yığın arasında %75 oranında kesişim sağlanmıştır. Birinci çalışma alanına kıyasla daha sık yığın oluşturmak gerekmiş ve ancak bu koşulda tüm noktalar bir yığın içerisinde en az bir sınıflandırma etiketi alabilmiştir.

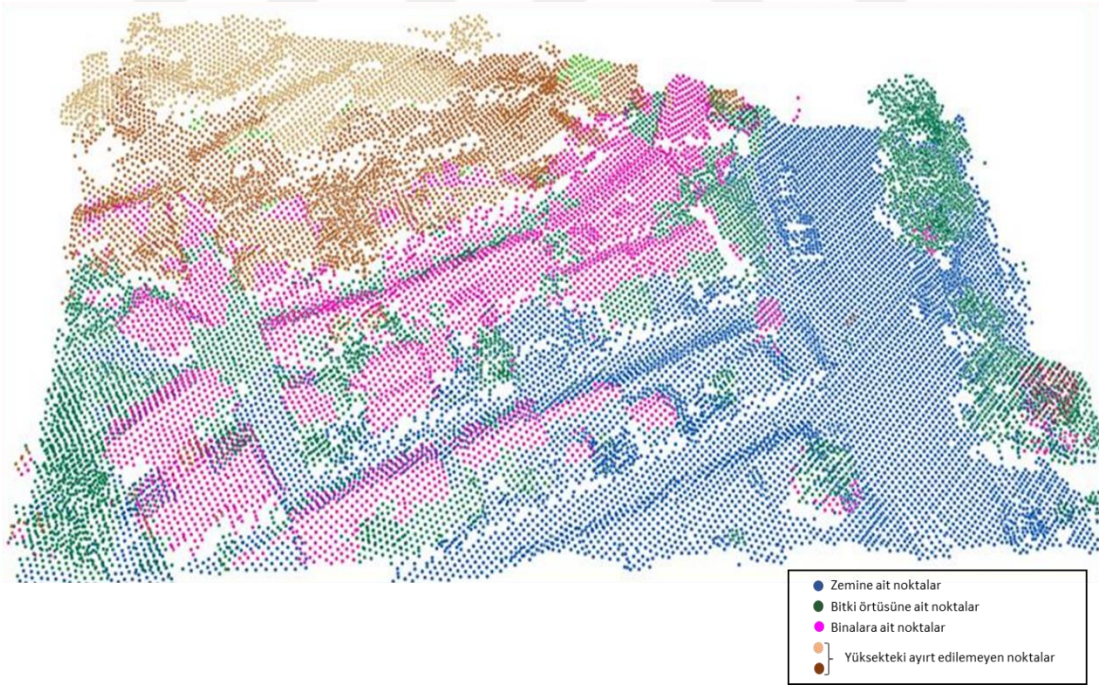
Ağ mimarisinde, Bölüm 5'te anlatıldığı gibi PointNet'in segmantasyon kısmına ait modelin bir kısmı kullanılmıştır. Tüm noktaların en az bir sınıflandırma etiketi aldığı son durumda, kesinlik, duyarlılık, f1-skor değerleri hesaplanmıştır. Sınıflandırma doğruluğu ve kayıp fonksiyon değerleri kontrol edilmiştir. Doğruluk ve kayıp fonksiyonlarının grafiklerine bakılarak tutarlılıkları değerlendirilmiştir.

7
BULGULAR

7.1 Kendini Düzenleyen Haritalar

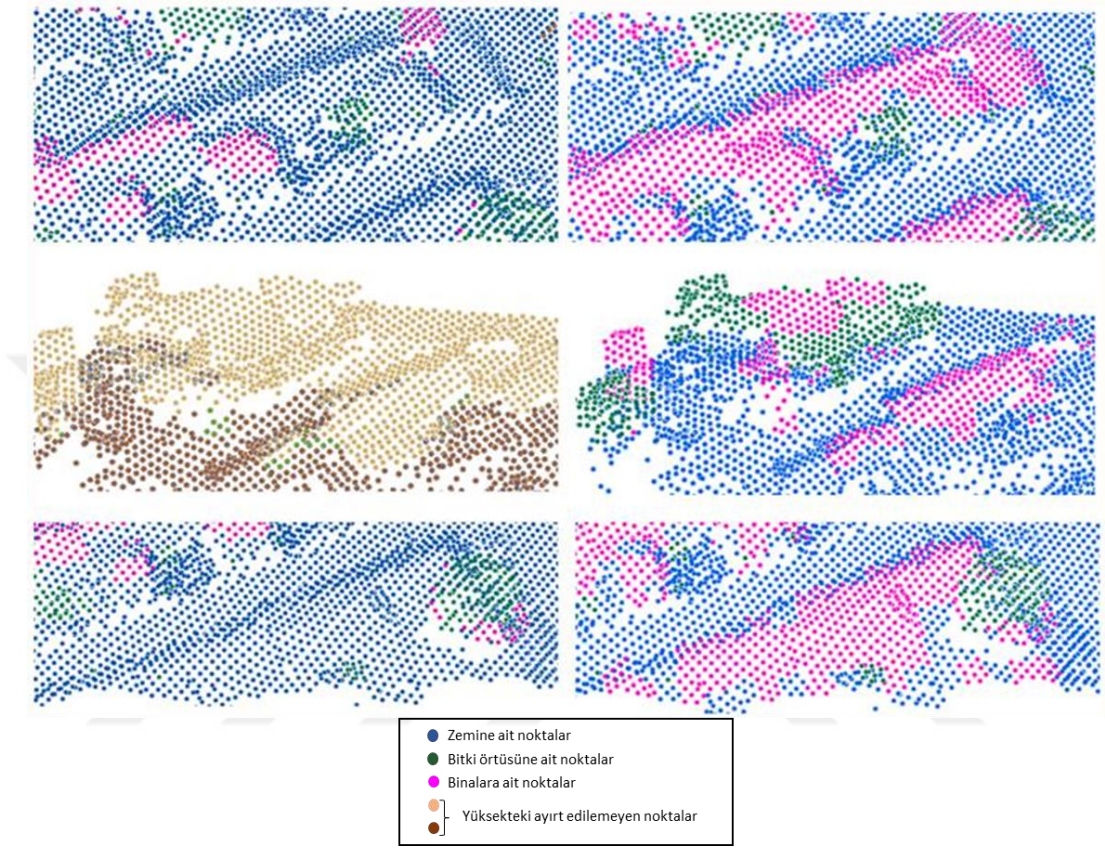
4.5.1. Birinci Çalışma Alanı

Genel anlamda en iyi performansı sağlayan kümelemede, pek çok bina noktasının zemin noktasından ayrıldığı görülmüştür. Ancak bina noktaları ile ortalama aynı yüksekliğe sahip bitki örtüsü noktalarının karıştığı, ayrıca bazı bina noktalarının zemin sınıflarında yer aldığı tespit edilmiştir. Şekil 7.1'de mavi renkte görülen zemin sınıfına ait noktalar ile bina noktalarını daha iyi ayrıştırmak ve şeklin sol üst köşesinde kahverengi renkte görülen eğimin arttığı yerde daha belirgin bir kümelenme oluşturmak için, söz konusu noktaları içeren altı küme, KDH ile üçer alt kümeye ayrılmıştır.

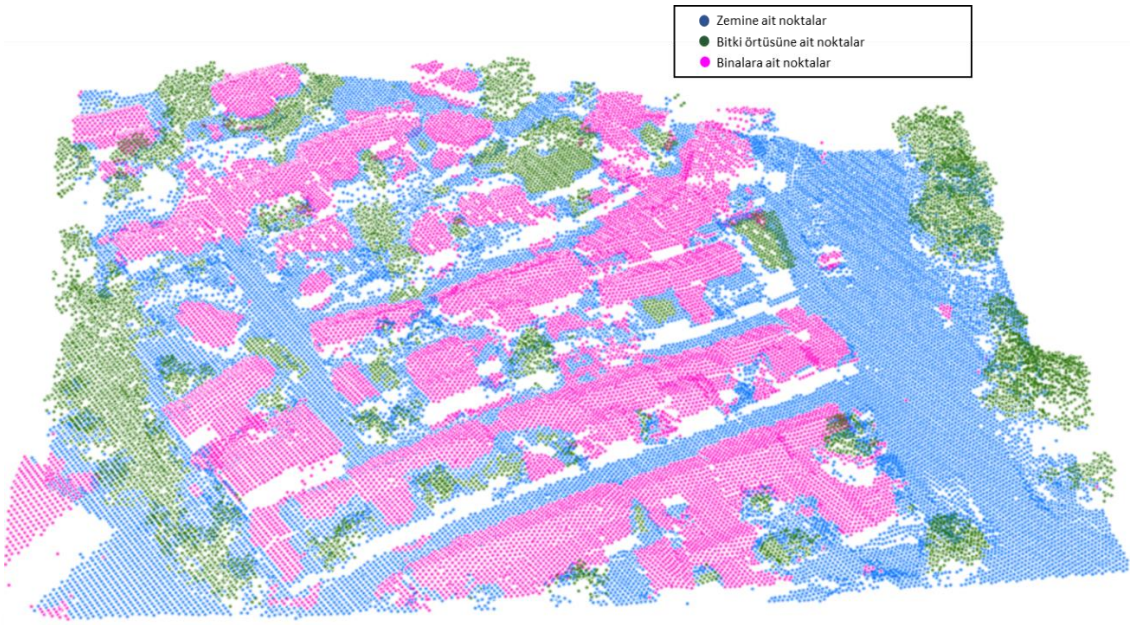


Şekil 7.1 Birinci çalışma alanına ilk uygulanan KDH sonrası sonuçlar

Bu aşamada 3x1 boyutlu nöron ağı yeterli olmuştur. İkinci aşama sonucunda bina olarak sınıflandırılmayan pek çok nokta, Şekil 7.2'de görüldüğü gibi zemin ve bitki örtüsü noktalarından ayrılmıştır.



Şekil 7.2 Birinci çalışma alanına ikinci kez KDH uygulandıktan sonra yakalanan bazı binalar (solda ilk KDH ile yakalanamayan, sağda ikinci KDH ile yakalananan)



Şekil 7.3 Birinci çalışma alanına ikinci kez uygulanan KDH sonrası sonuçlar

Sistematik bir şekilde az sayıda küme oluşturularak en etkili nokta sınıflandırmasının belirlenmesi amaçlanmıştır. Bu nedenle en yüksek sınıflandırma doğruluna en az sayıda küme ile ulaşılmak istendiğinden, hata tolerans değeri %20 ile sınırlı tutulmuştur. Daha düşük tolerans değerleri, nöron veya küme sayısını arttırmış ve görsel olarak yapılan kümelerin birleştirilmesi işlemini karmaşık hale getirmiştir.

Sınıflandırma doğruluğunu belirleyebilmek için uydu ve sokak görüntülerinden yardım alınarak öncelikle tüm noktalar manuel olarak yüksek hassasiyet ile sınıflandırıldı. Toplam 46438 noktanın 17821'i bina, 10865'i bitki örtüsü, 17752'si zemine ait noktalardır. Bu sınıflandırma sonuçları ile kendini düzenleyen haritalar sonucunda elde edilen sonuçlar kıyaslanmıştır. Tablo 7.1'de görüldüğü gibi bina, bitki örtüsü ve zemin noktaları içeren bir sınıflandırma matrisi hazırlanmıştır. Matrise göre bina-bina, bitki örtüsü-bitki örtüsü ve zemin-zemin kesişim hücreleri doğru sınıflandırılan noktaları göstermektedir. Diğer hücrelerdeki nokta sayıları ise zemine karışan bina noktaları, binaya karışan bitki örtüsü noktalarını veya bitki örtüsüne karışan bina noktaları gibi diğer yanlış sınıflandırılan nokta sayılarını göstermektedir. Tablo 7.1'de bina, bitki örtüsü ve zemin olmak üzere üç sınıfa ait sınıflandırma doğruluk yüzdeleri gösterilmiştir. Tablo 7.2'de ise, noktaların filtrelenmesi ile ilgili Tip-1, Tip-2 ve toplam hata oranı gösterilmiştir.

Tablo 7.1 Birinci çalışma alanı KDH ile sınıflandırma hata tablosu

		Kendini Düzenleyen Haritalar			
	Sınıflandırma	Bina	Bitki Örtüsü	Zemin	Doğruluk
Referans	Bina	**13068**	1123	3630	0,73
	Bitki Örtüsü	82	**8840**	1943	0,81
	Zemin	1631	418	**15703**	0,88
				Genel doğruluk:	**0,81**

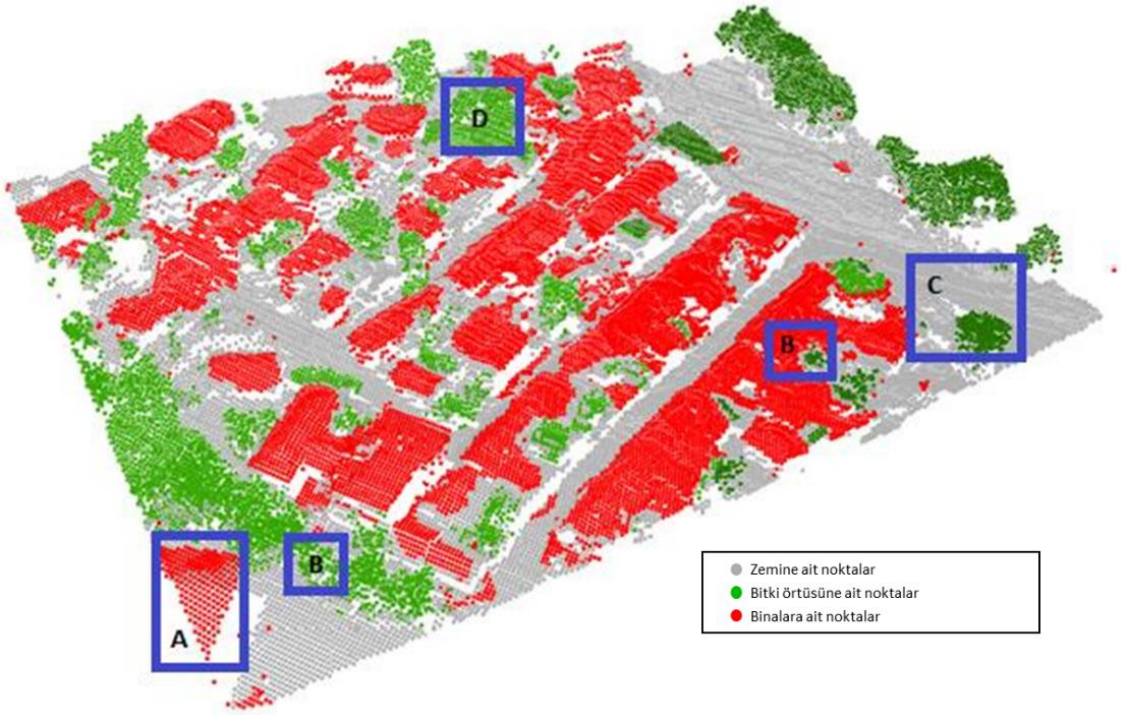
Tablo 7.1'de diyagonal elemanlar doğru sınıflandırılan noktaları gösterirken, sütundaki diğer elemanlar hatalı sınıflandırılmış noktaları göstermektedir. Toplamda 46438 noktanın 37611'i doğru sınıflandırılmıştır. KDH yöntemi ile sınıflandırmanın genel başarısı %80.99'dur. Buna göre, toplam 17821 adet bina noktasının 13068'i doğru sınıflandırılırken, 1123'ü bitki örtüsü, 3630'u zemin olarak yanlış sınıflandırılmıştır. Bina noktalarının çıkarımı %73.33 başarılıdır. Sonuçlar, bir bina çıkarımı yöntemi olarak da yöntemin geliştirilebileceğini göstermiştir. Bitki örtüsüne ait toplam 10865 adet noktanın 8840'ı doğru sınıflandırılırken, 82'si bina, 1943'ü zemin olarak yanlış sınıflandırılmıştır. Bitki örtüsünün çıkarımı %81.36 başarılıdır. Günümüzde bitki örtüsüne ait noktaların analizinde tam dalga boyu formlu sistemler kullanılmaktadır. Ayrık dönüşlü sistemler, bitki örtüsüne ait noktaların çıkarımında tam dalga boyu formlu sistemlere göre daha zayıf kalsa da bu çalışmadaki performansı iyidir. Tablo 7.1'de, toplam 17752 adet zemin noktasının 15703'ü doğru sınıflandırılırken, 1631'i bina, 418'i bitki örtüsü olarak yanlış sınıflandırılmıştır. Zemin noktalarının çıkarımı %88.46'dır. Uygulanan yöntemin filtreleme performansı iyi düzeydedir.

Tablo 7.2 Birinci çalışma alanı KDH ile filtreleme hata tablosu

		Kendini Düzenleyen Haritalar			
	Filtreleme	Zemine Ait Noktalar	Zemine Ait Olmayan Noktalar		
Referans	Zemine Ait Noktalar	15703 (a)	2049 (b)	Tip-1 Hatası	2049 / (15703+2049) = **%11.54**
	Zemine Ait Olmayan Noktalar	5573 (c)	23113 (d)	Tip-2 Hatası	5573 / (5573+23113) = **%19.43**
				Toplam Hata	(2049+5573) / 46438 = **%16.41**

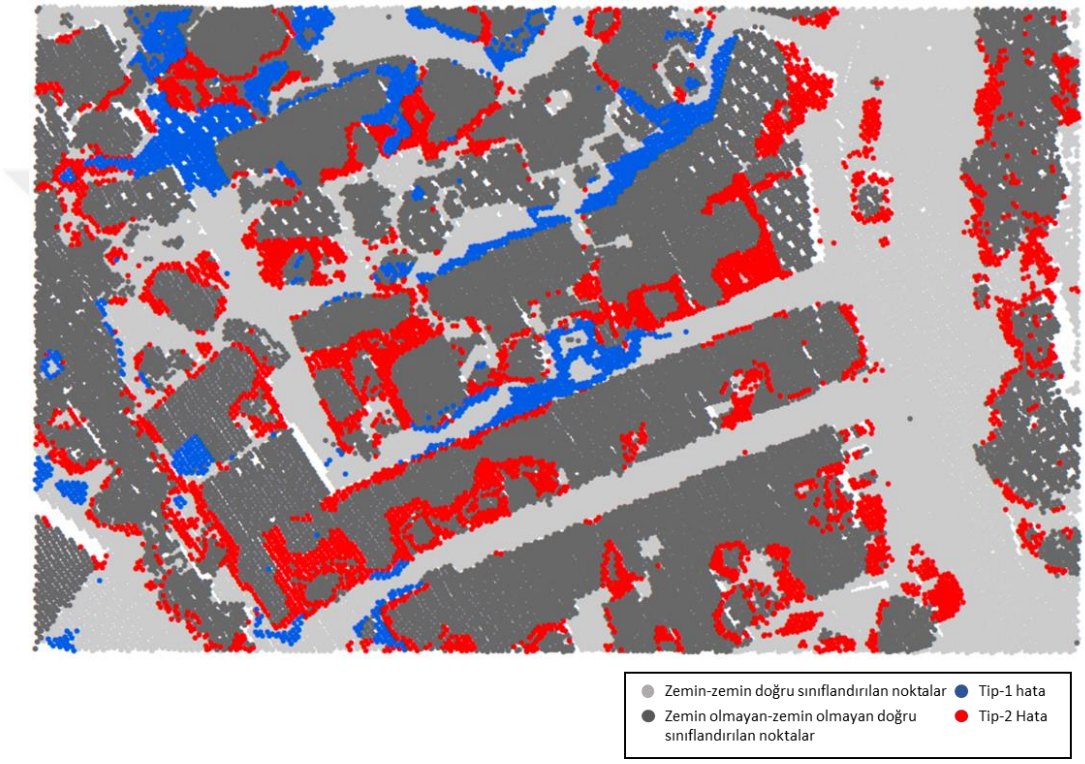
Tablo 7.2'de, Lidar nokta bulutunun denetimsiz bir yapay sinir ağı olan KDH yöntemiyle filtrelenmesi sonucu, zemine ait olmayan nokta olarak yanlış sınıflandırılmış zemine ait noktalardaki hatayı gösteren Tip-1 hatası %11.54'tür. Bu oran ile Tablo 1'de görülen zemine ait noktaların çıkarımı oranının toplamı 100'e eşittir. Zemine ait nokta olarak yanlış sınıflandırılmış zemine ait olmayan noktalardaki hatayı gösteren Tip-2 hatası %19.43'tür. Toplam filtreleme hatası ise %16.41 olarak bulunmuştur.

Hata tolerans değerinin %20 eşik değerden yüksek olduğu kümelere alt kümeler oluşturmak amacıyla ikinci kez KDH uygulanmış ve tüm kümelerde istenilen doğruluk değerlerine ulaşılmıştır. Sonuçlar görsel olarak değerlendirildiğinde Şekil 7.4'te A kutucuğunda yer alan küme üç alt kümeye ayrılarak zemine karışan bina noktaları doğru sınıflandırılmıştır. B kutucuklarında yer alan kümeler üçer alt kümeye ayrılarak zemine karışan bitki örtüsü doğru sınıflandırılmıştır. C kutucuğunda yer alan küme üç alt kümeye ayrılarak bitki örtüsüne karışan zemin noktaları doğru sınıflandırılmıştır. Ancak D kutucuğunda yer alan bitki örtüsüne karışan bina noktalarının sınıflandırılabilmesi için daha fazla sayıda alt küme oluşturulması gerektiği gözlemlenmiştir.



Şekil 7.4 KDH yöntemiyle sınıflandırılan zemin, bina ve bitki örtüsü noktaları

Şekil 7.5'te doğru sınıflandırılan zemin noktaları açık gri renk ile, doğru sınıflandırılan zemine ait olmayan noktalar koyu gri renk ile, Tip-1 hatası mavi renk ile ve Tip-2 hatası kırmızı renk ile gösterilmiştir. Genellikle hatalar eğimin arttığı yerde, binalar arasında bulunan bahçe duvarlarından ve çatılardan uzanan saçaklardan kaynaklanmaktadır.



Şekil 7.5 Birinci alanın KDH ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi

4.5.2. İkinci Çalışma Alanı

Genel sınıflandırma doğruluğunun birinci alışma alanına ile hemen hemen aynı olduğu fakat filtreleme sonuçları biraz daha iyi olduğu gözlenmiştir. Ayrıca, birinci çalışma alanına göre bina ve zemin noktalarının daha iyi doğrulukla ayrıştırıldığı, ancak bitki noktalarının yeterince iyi ayrıştırılamadığı görülmüştür.

Tablo 7.3 İkinci çalışma alanı KDH ile sınıflandırma hata tablosu

		Kendini Düzenleyen Haritalar			
	Sınıflandırma	Bina	Bitki Örtüsü	Zemin	Doğruluk
Referans	Bina	**17648**	939	3532	0,80
	Bitki Örtüsü	2591	**6525**	2858	0,54
	Zemin	465	792	**23375**	0,95
				Genel Doğruluk:	**0,81**

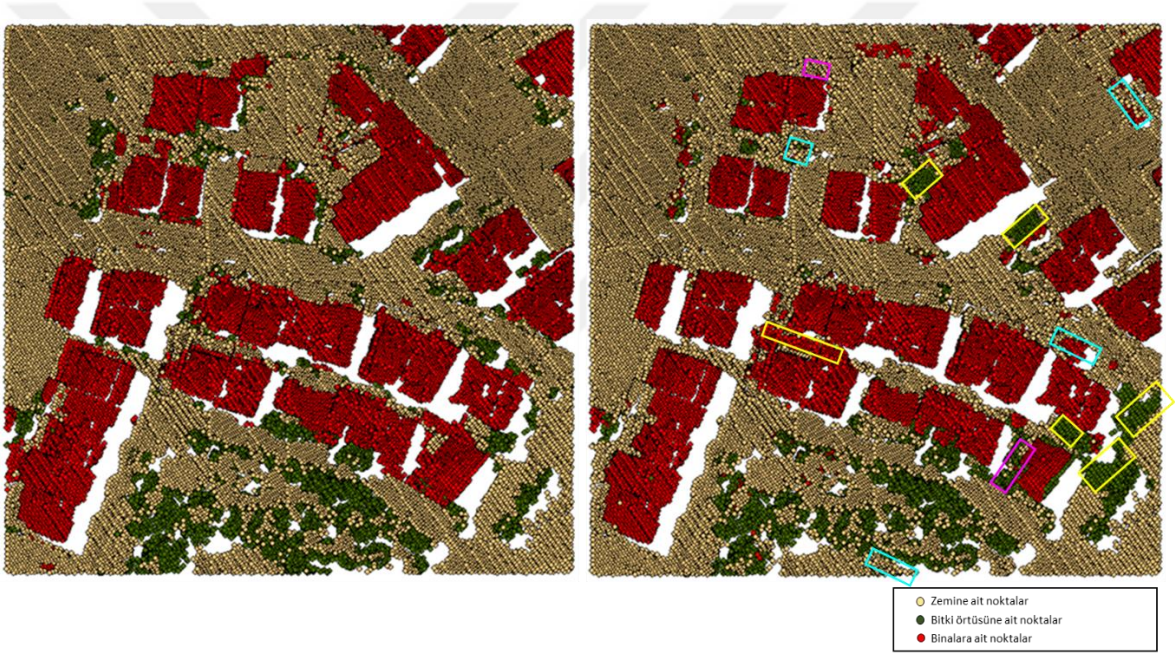
Tablo 7.3'e göre, toplamda 58725 noktanın 47548'i doğru sınıflandırılmıştır. KDH yöntemi ile sınıflandırmanın genel başarısı %80.97'dir. Buna göre, toplam 22119 adet bina noktasının 17648'i doğru sınıflandırılırken, 939'u bitki örtüsü, 3532'si zemin olarak yanlış sınıflandırılmıştır. Bina noktalarının çıkarımı %79.79 başarılıdır. Bitki örtüsüne ait toplam 11974 adet noktanın 6525'i doğru sınıflandırılırken, 2591'i bina, 2858'i zemin olarak yanlış sınıflandırılmıştır. Bitki örtüsünün sınıflandırmasının doğruluğu %54.49 olup, beklenen başarı sağlanamamıştır. Ancak zemin noktalarının çıkarılma doğruluğunun yüksek olduğunu hesaba katarsak, bu pek çok bitki örtüsü noktasının kendisiyle aynı yükseklikte ve kendisine çok yakın olan bina noktalarıyla karıştığı anlamına gelmektedir. Aynı şekilde zeminden çok yüksek olmayan bitki örtüsünün (low vegetation) de zemin noktalarına karıştığı çıkarımı yapılabilmektedir. Toplam 24632 adet zemin noktasının 23375'i doğru sınıflandırılırken, 465'i bina, 792'si bitki örtüsü olarak yanlış sınıflandırılmıştır. Zemin noktalarının çıkarımı %94.90'dır. Uygulanan yöntemin zemin noktası çıkarım performansı oldukça iyi düzeydedir.

Tablo 7.4 İkinci çalışma Alanı KDH ile filtreleme hata tablosu

		Kendini Düzenleyen Haritalar			
	Filtreleme	Zemine Ait Noktalar	Zemine Ait Olmayan Noktalar		
Referans	Zemine Ait Noktalar	23375 (a)	1257 (b)	Tip-1 Hatası	1257 / (23375+1257) = **%5.10**
	Zemine Ait Olmayan Noktalar	6390 (c)	27703 (d)	Tip-2 Hatası	6390 / (6390+27703) = **%18.74**
				Toplam Hata	(1257+6390) / 58725 = **%13.02**

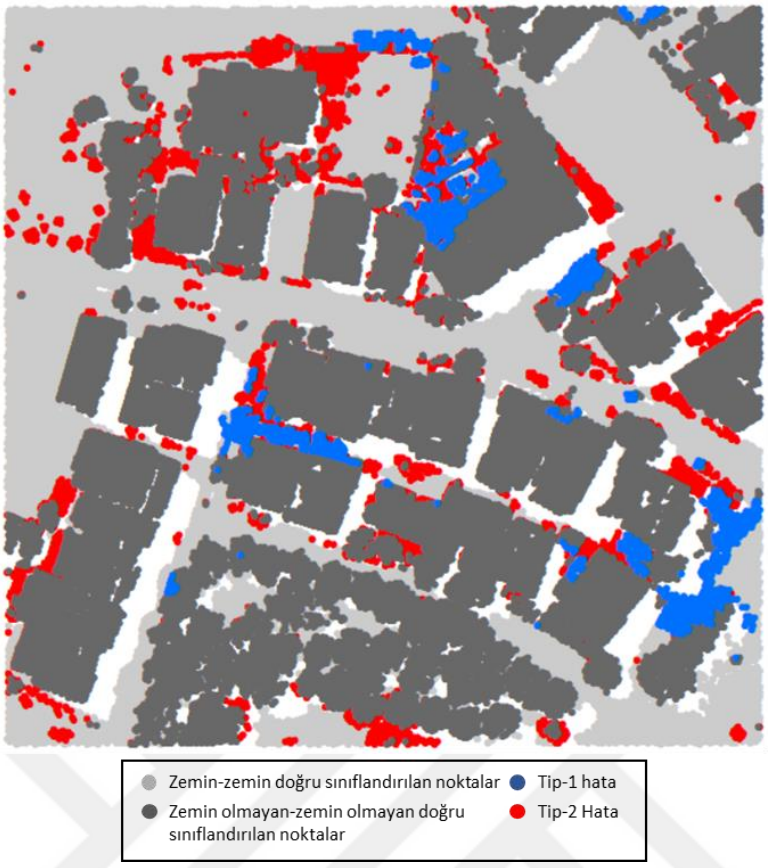
İkinci çalışma alanı nokta bulutunun filtrelenmesi sonucu Tip-1 hatası %5.10, Tip-2 hatası %18.74 hesaplanmıştır. Toplam filtreleme hatası %13.02 iken, filtreleme doğruluğu 86,98'dir.

Şekil 7.6'da sınıflandırma sonuçları görselleştirilmiştir. Sağdaki görsel referans sınıflandırmaya, soldaki görsel ise KDH sınıflandırması sonucuna aittir. Mavi kutular içerisindeki alanlarda bitki örtüsü çıkarımı, sarı renkli kutular içerisindeki alanlarda zemin noktası çıkarımı, pembe renkli kutular içerisinde ise bina nokta çıkarımı gerçekleşememiştir.



Şekil 7.6 İkinci çalışma alanının KDH ile son sınıflandırma sonuçları

Şekil 7.7'de doğru filtrelenen zemin noktaları açık gri renk ile, doğru filtrelenen zemine ait olmayan noktalar koyu gri renk ile, Tip-1 hatası mavi renk ile ve Tip-2 hatası kırmızı renk ile gösterilmiştir.

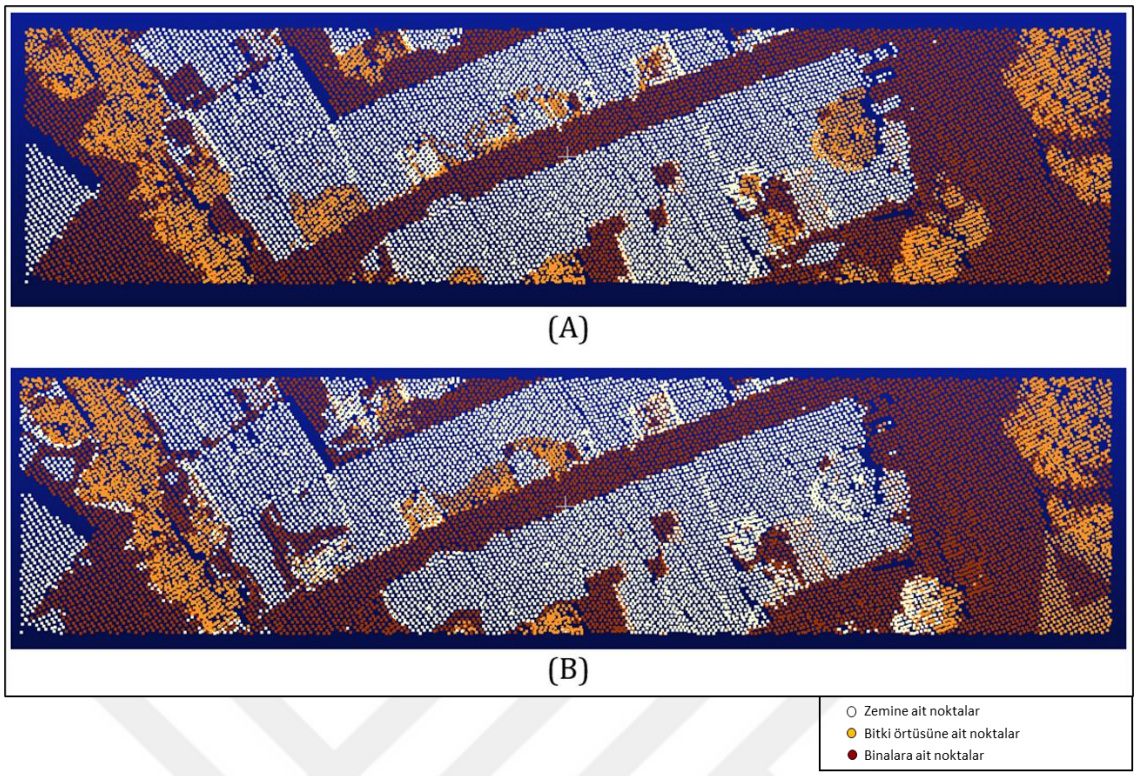


Şekil 7.7 İkinci çalışma alanının KDH ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi

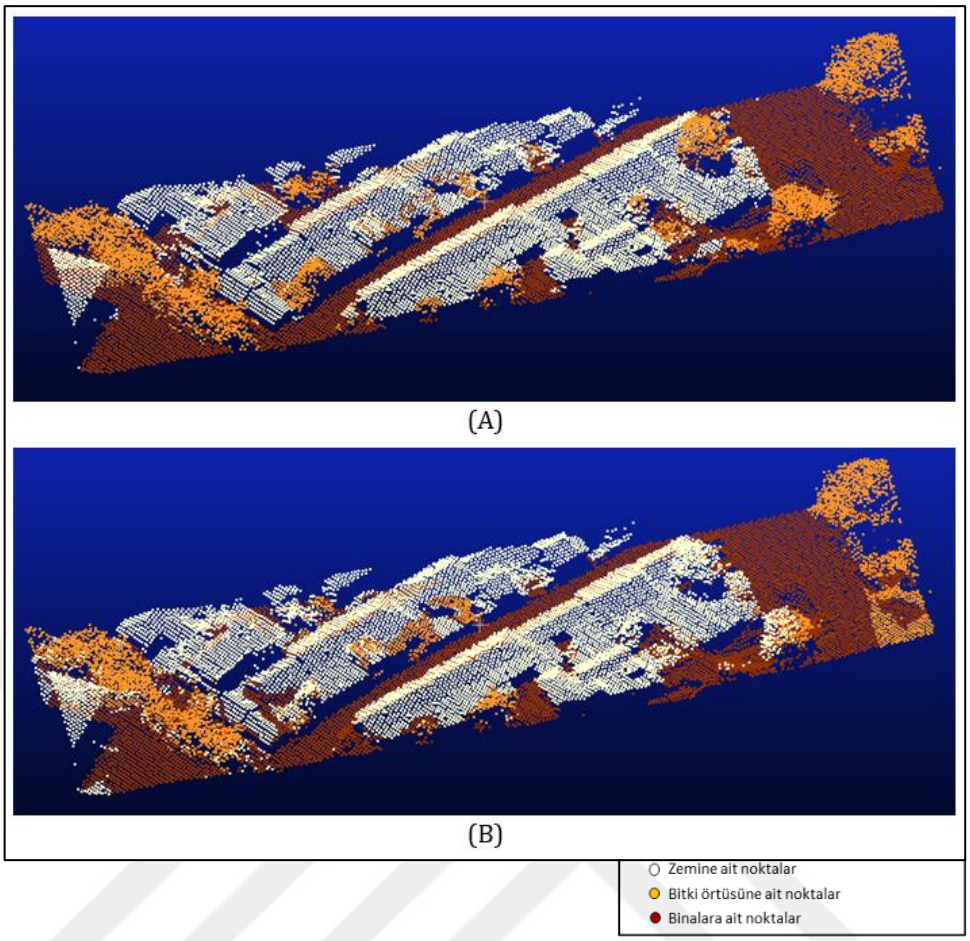
7.2 Rastgele Orman Algoritması

4.5.3. Birinci Çalışma Alanı

Rasgele Orman ile test verisi zemin ve zemine ait olmayan noktalar olarak filtrelenmiş ve bina, bitki örtüsü, zemin noktaları olmak üzere sınıflandırılmıştır. Şekil7.8 A görüntüsünde referans sınıflandırma, B görüntüsünde ise rastgele orman ile sınıflandırma sonucuna göre noktaların görselleştirilmesi verilmiştir. Genel itibariyle bina ve zemin noktaları çıkarımı, bitki örtüsü çıkarımına göre daha başarılı olmuştur. Sağ en alt köşede bazı zemin noktalarının bitki örtüsü olarak yanlış sınıflandırıldığı ve bazı bitki noktalarının bina noktalarına karıştığı gözlenmiştir. Şekil 7.9 ise, Şekil 7.8'in perspektifli halidir.



Şekil 7.8 RO algoritması ile sınıflandırma sonuçları: (A) manuel, (B) RO



Şekil 7.9 RO algoritması ile sınıflandırma sonuçları (perspektif) (A)manuel (B)RO

Sınıflandırma sonuçları hata matrisiyle değerlendirilmiştir. 17066 tane nokta içeren test veri seti için oluşturulan Tablo 7.5'teki hata matrisine göre zemin sınıflandırma doğruluğunun %88.26, bitki örtüsü sınıflandırma doğruluğunun %66.54, ve bina sınıflandırma doğruluğunun %90.89 olduğu tespit edilmiştir. Genel sınıflandırma doğruluğu ise %84.27'dir. Burada da bitki örtüsü noktalarının sınıflandırma doğruluğu diğer nesnelere göre daha düşük olduğu gözlenmiştir.

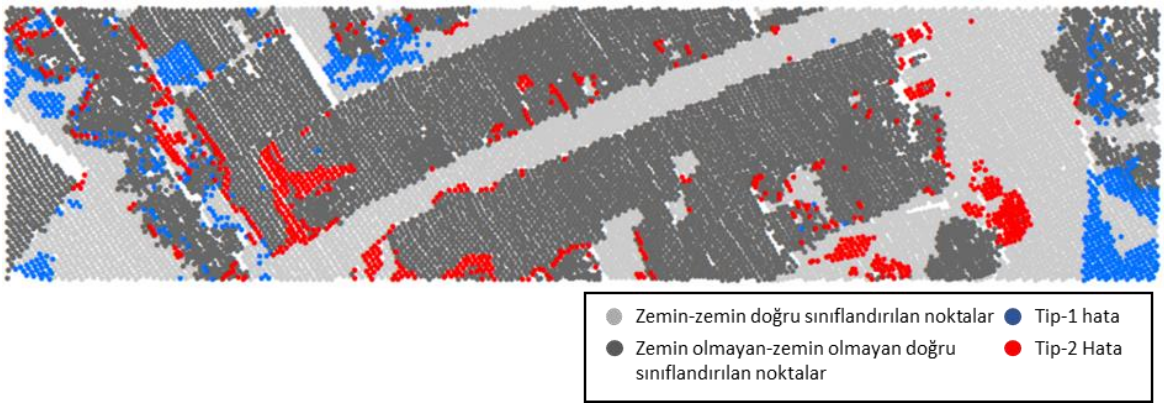
Tablo 7.5 Birinci çalışma alanı RO algoritması ile sınıflandırma hata tablosu

		Rastgele Orman Algoritması			
	Sınıflandırma	Bina	Bitki Örtüsü	Zemin	Doğruluk
Referans	Bina	**6159**	158	459	0,91
	Bitki Örtüsü	924	**2631**	399	0,67
	Zemin	300	444	**5592**	0,88
				Genel Doğruluk:	**0,84**

Filtreleme sonuçları ise, sınıflandırma sonuçlarına nispeten daha iyidir**.** Tablo 7.6'da görüldüğü üzere Tip-1 hatası %11.74, Tip-2 hatası %8 ve toplam hata %9.39 olarak hesaplanmıştır. Filtreleme genel doğruluğu %90,61'dir.

Tablo 7.6 Birinci çalışma alanı RO algoritması ile filtreleme hata tablosu

Rastgele Orman Algoritması					
	Filtreleme	Zemine Ait Noktalar	Zemine Ait Olmayan Noktalar		
Referans	Zemine Ait Noktalar	5592 (a)	744 (b)	Tip-1 Hatası	744 / (5592+744) = **%11.74**
	Zemine Ait Olmayan Noktalar	858 (c)	9872 (d)	Tip-2 Hatası	858 / (858+9872) = **%8**
				Toplam Hata	(744+858) / 17066 = **%9.39**



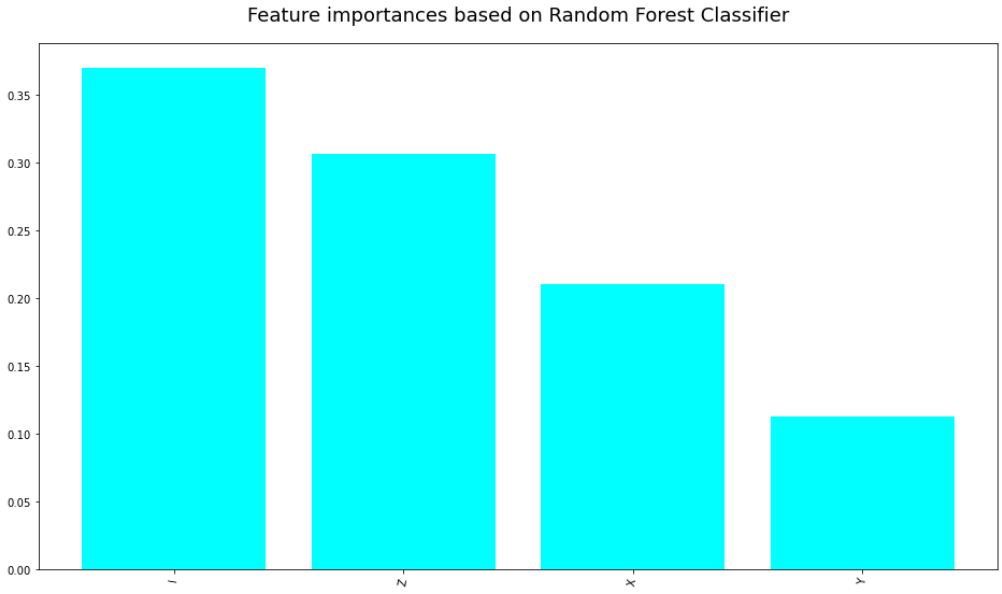
Şekil 7.10 Birinci alanın RO algoritması ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi

Sınıflandırma sonuçları için ayrıca kesinlik, duyarlılık ve f1-skor ölçütleri de hesaplanmıştır. Tablo7.7'de gösterilen 1, 2, 3 sayıları sırasıyla zemin, bitki örtüsü ve bina sınıflarını temsil etmektedir. Son sütunda ise her sınıftan toplam kaç nokta olduğu gösterilmektedir.

Tablo 7.7 Birinci çalışma alanı RO algoritması ile sınıflandırma doğruluk ölçütleri *(1, 2 ve 3 etiketleri sırasıyla zemin, bitki örtüsü ve binayı temsil etmektedir. Precision, recall, f1-score, support, accuracy sırasıyla kesinlik, duyarlılık, f1-skor, nokta sayısını ve doğruluğu göstermektedir.)*

	precision	recall	f1-score	support
1	0.87	0.88	0.87	6336
2	0.81	0.67	0.73	3954
3	0.83	0.91	0.87	6776
accuracy			0.84	17066
macro avg	0.84	0.82	0.83	17066
weighted avg	0.84	0.84	0.84	17066

Rastgele orman algoritmasında *x, y, z* ve yasıma değerlerinin sınıflandırmayı hangi derecede etkilediğini gösteren grafik ise Şekil 7.11'de gösterilmiştir. Öncesinde hiçbir ağırlık veya eşik parametresi kullanılmamasına rağmen, sınıflandırmada en önemli etkiyi yansıma değeri, ardından yükseklik değeri sağlamıştır.

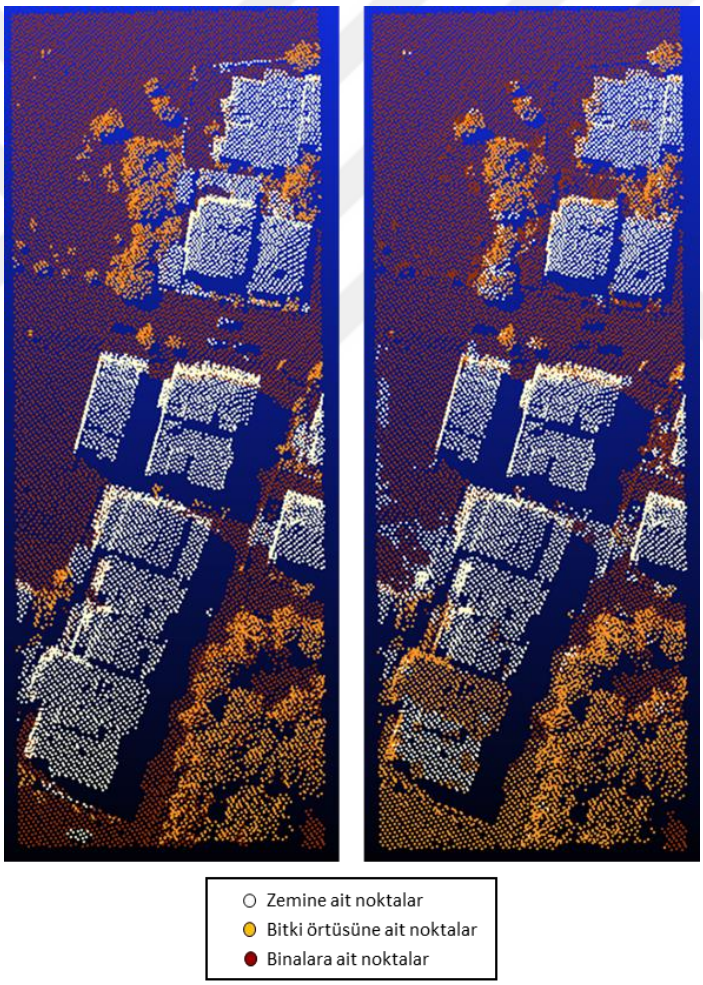


Şekil 7.11 Birinci çalışma alanı için *x,y,z,i* değerlerinin sınıflandırmayı etkileme oranları

4.5.4. İkinci Çalışma Alanı

Birinci çalışma alanına uygulanan aynı metodoloji aynı yazılımlar ile ikinci çalışma alanına da uygulanmıştır. Şekil 7.12'de A görüntüsünde referans sınıflandırma, B görüntüsünde ise rastgele orman ile sınıflandırma sonucuna göre noktaların görselleştirilmesi verilmiştir.

Şeklin üst kısımlarda sınıflandırmanın daha iyi olduğu görülmüştür. En alt kısımda ise bazı bina noktalarının ve zemin noktalarının bitki örtüsü noktaları olarak yanlış sınıflandırıldığı gözlenmiştir. Buna, o alandaki yansıma değerlerinin birbirine çok yakın olması sebep olarak gösterilebilir.



(A) (B)

Şekil 7.12 İkinci çalışma alanının RO algoritması ile sınıflandırma sonuçları (A)manuel (B)RO

Toplam 20462 nokta içeren test alanının sınıflandırılmasına ait hata tablosuna göre zemin noktalarının %86.23'ü, bitki örtüsü noktalarının %75,56'sı, bina noktalarının ise %75.40'ı doğru sınıflandırılmıştır. Genel sınıflandırma doğruluğu %80.5 olarak tespit edilmiştir.

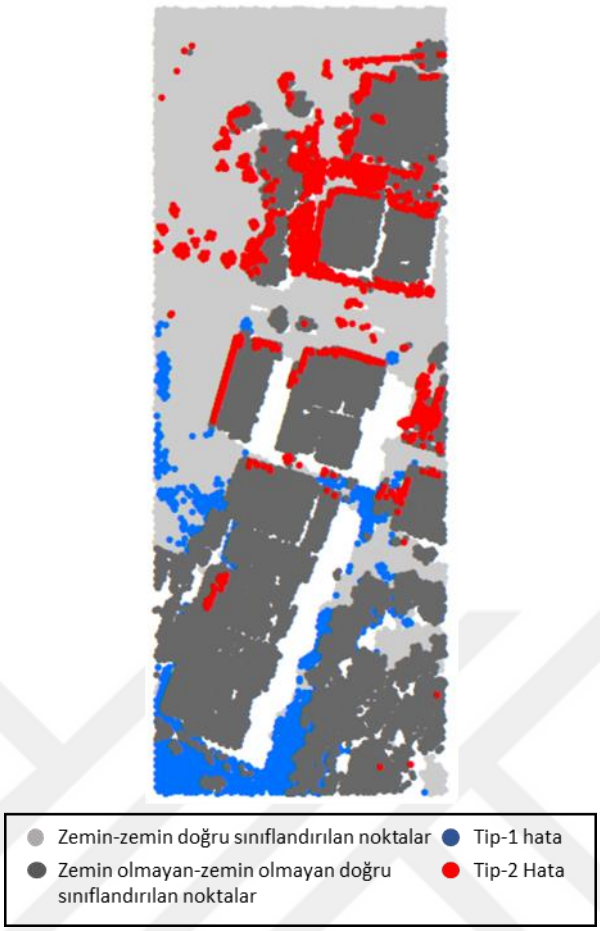
Tablo 7.8 İkinci çalışma alanı RO algoritması ile sınıflandırma hata tablosu

	Rastgele Orman Algoritması				
	Sınıflandırma	Bina	Bitki Örtüsü	Zemin	Doğruluk
Referans	Bina	**5366**	998	753	0,75
	Bitki Örtüsü	303	**2739**	583	0,76
	Zemin	416	922	**8382**	0,86
				Genel Doğruluk:	**0,81**

Filtreleme genel doğruluğu %86.93 olarak hesaplanmıştır. Bu alanda da filtrelemenin, sınıflandırma sonucuna göre biraz daha iyi olduğu gözlemlenmiştir. Tip-1 hata %13.77, Tip-2 Hata %12.44 ve toplam hata %13.07 olarak hesaplanmıştır.

Tablo 7.9 İkinci çalışma alanı RO algoritması ile filtreleme hata tablosu

	Rastgele Orman Algoritması				
	Filtreleme	Zemine Ait Noktalar	Zemine Ait Olmayan Noktalar		
Referans	Zemine Ait Noktalar	8382 (a)	1338 (b)	Tip-1 Hatası	1338 / (8382+1338) = **%13.77**
	Zemine Ait Olmayan Noktalar	1336 (c)	9406 (d)	Tip-2 Hatası	1336 / (1336+9406) = **%12.44**
				Toplam Hata	(1338+1336) / 20462 = **%13,07**

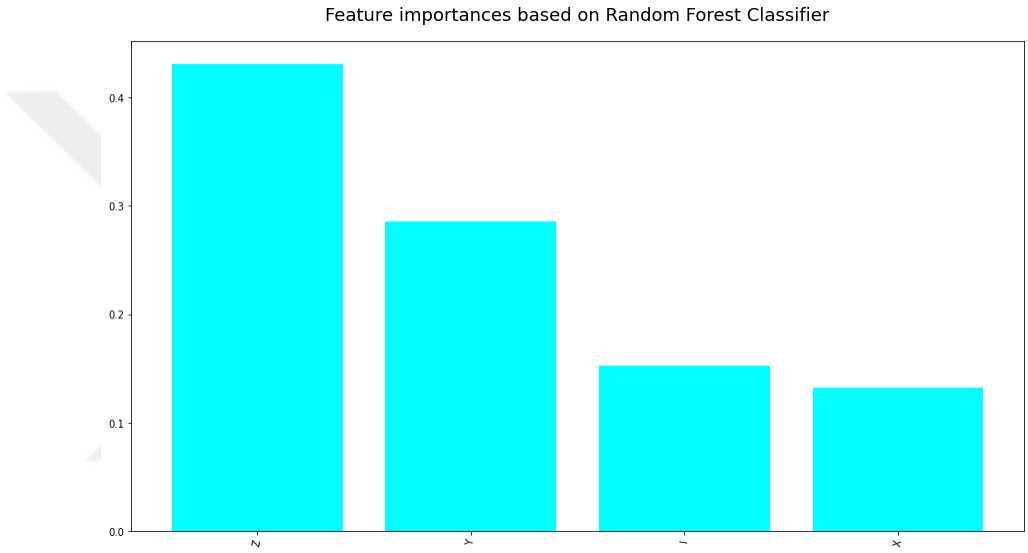


Şekil 7.13 İkinci alanın RO algoritması ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi

Sınıflandırma sonuçları için kesinlik, duyarlılık ve f1-skor ölçütleri de önceki alanda olduğu gibi tespit edilmiştir. Sınıflandırmada en önemli yükseklik değeri ardından, y değeri sağlamıştır. Yansıma değerinin yüksek bir öneme sahip olmaması yukarıda da belirtildiği gibi birbirine çok yakın değerlere sahip olmasından kaynaklanmaktadır. Bu test alanına göre yansıma değerinin sınıflandırma için ayırt edici olma özelliğinin düşük kaldığı gözlenmiştir.

Tablo 7.10 İkinci çalışma alanı RO algoritması ile sınıflandırma doğruluk ölçütleri *(1, 2 ve 3 etiketleri sırasıyla zemin, bitki örtüsü ve binayı temsil etmektedir.) (precision, recall, f1-score, support, accuracy sırasıyla kesinlik, duyarlılık, f-1 skor, nokta sayısını ve doğruluğu göstermektedir.)*

	precision	recall	f1-score	support
1	0.86	0.86	0.86	9720
2	0.59	0.75	0.66	3625
3	0.89	0.75	0.81	7117
accuracy			0.81	20462
macro avg	0.78	0.79	0.78	20462
weighted avg	0.82	0.81	0.81	20462



Şekil 7.14 İkinci çalışma alanı için x,y,z,i değerlerinin sınıflandırmayı etkileme oranları

7.3 Derin Öğrenme

Birinci ve ikinci çalışma alanları için Rastgele orman algoritmasında kullanılan eğitim ve test verileri aynıdır. Derin öğrenmede bu girdiler yığınlar haline getirilerek kullanılmıştır. Sınıflandırmada ise kendini düzenleyen haritalar ve rastgele orman algoritmasından daha iyi sonuçlar vermiştir. Ancak bu iki algoritma sonuçları 1 dakikadan az bir sürede verirken, derin öğrenmede bu süre birinci çalışma alanı için 1 buçuk saati ve ikinci çalışma alanı için neredeyse 6 saati bulmuştur (Bilgisayar özellikleri: i7-6700HQ CPU @ 2.60GHz, 2592 Mhz, 4 Çekirdek,

8 Mantıksal İşlemci). Ayrıca girdi verisinin hazırlığı, ağ mimarisinin oluşturulması ve parametrelerin belirlenmesi biraz daha komplike ve zaman alıcıdır.

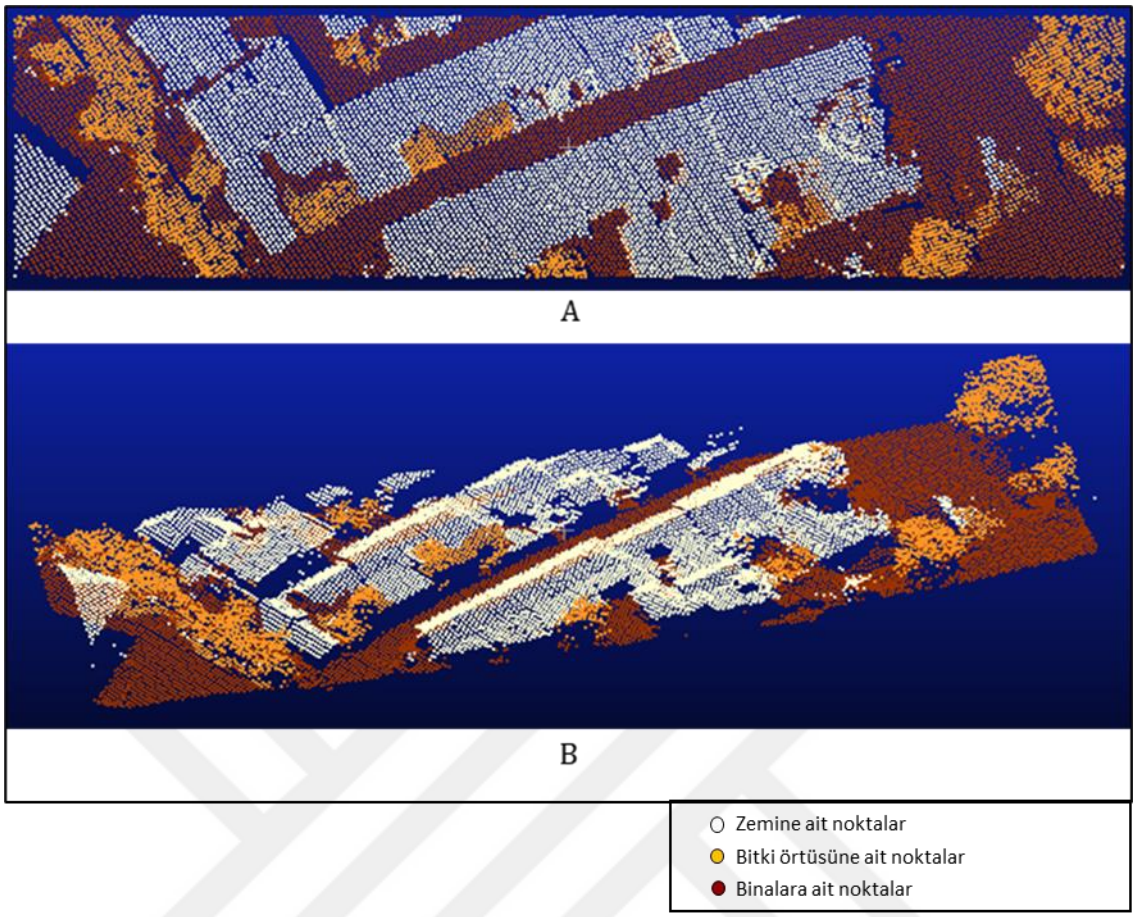
İkinci çalışma alanının birinci çalışma alanından daha uzun sürede sınıflandırılmasının temel sebebi ise yığın merkezlerinin çok daha sıklaştırılmış olmasıdır. İkinci çalışma alanındaki yığın sayısı, birinci çalışma alanının neredeyse 3.5 katıdır.

Daha sonra sınıflandırma doğruluğunun artması beklentisiyle birinci çalışma alanında da yığın sıklaştırmasına gidilmiş; ancak sınıflandırma sonucunda pek bir değişiklik gözlenmemiştir. Sadece çok uzun ve gerek duyulmayan bir eğitim süreci gerçekleşmiştir. Böylelikle, tüm noktalar kapsandığı sürece ve eğitim ve test verisi değişmediği sürece yığın sıklığının sınıflandırma doğruluğunu pek arttırmadığı ortaya çıkmıştır.

4.5.5. Birinci Çalışma Alanı

Ağın eğitimi yaklaşık 92 dakika sürmüştür. Test alanının genel sınıflandırma doğruluğu %87.96 olarak hesaplanmıştır. Zemin noktaların çıkarımı %94.90, bitki örtüsü noktalarının çıkarımı %75.77, bina noktalarının çıkarımı %88.58 doğrulukla gerçekleşmiştir.

Şekil 7.15'te birinci çalışma alanına ait test verisinin tahmin edilen sınıf etiketleriyle görselleştirilmesi verilmiştir. A görünümü nokta bulutunun tepeden, B görünümü ise perspektifli görünümüdür. Tablo 7.11'de ise sınıflandırma hata matrisi verilmiştir.



Şekil 7.15 Birinci çalışma alanının derin öğrenme ile sınıflandırma sonucu (A) tepeden görünüm (B) perspektif görünüm

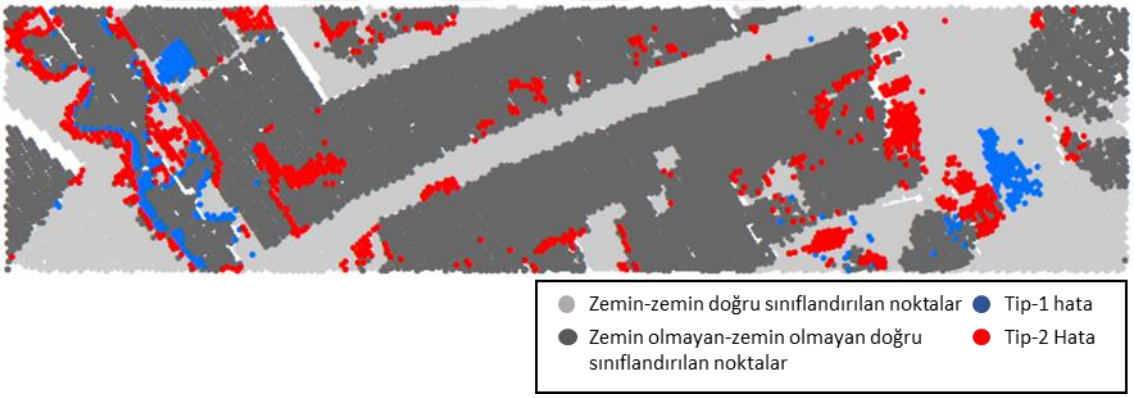
Tablo 7.11 Birinci çalışma alanı Derin Öğrenme ile sınıflandırma hata tablosu

		Derin Öğrenme			
	Sınıflandırma	Bina	Bitki Örtüsü	Zemin	Doğruluk
Referans	Bina	**6002**	205	569	0,89
	Bitki Örtüsü	449	**2996**	509	0,76
	Zemin	109	214	**6013**	0,95
				Genel Doğruluk:	**0,88**

Tip-1 hatası %6.04, Tip-2 hatası %10.05, toplan hata ise %8.21 olarak hesaplanmıştır. Filtreleme doğruluğu %91.79'dur. Tablo 7.12 ile filtreleme hata matrisi, Şekil 7.16 ile de filtreleme sonucu hesaplanan Tip-1 ve Tip-2 hatalarının gösterimi verilmiştir.

Tablo 7.12 Birinci çalışma alanı Derin Öğrenme ile filtreleme hata tablosu

Derin Öğrenme					
	Filtreleme	Zemine Ait Noktalar	Zemine Ait Olmayan Noktalar		
Referans	Zemine Ait Noktalar	6013 (a)	323 (b)	Tip-1 Hatası	383 / (6013+323) = **%6.04**
	Zemine Ait Olmayan Noktalar	1078 (c)	9652 (d)	Tip-2 Hatası	1078 / (1078+9652) = **%10.05**
				Toplam Hata	(323+1078) / 17066 = **%8.21**



Şekil 7.16 Birinci alanın derin öğrenme ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi

Sınıflandırma sonuçlarının kesinlik, duyarlılık ve f1-skor gibi doğruluk kriterleri hesaplanmıştır. Eğitimin tamamlanma süresiyle birlikte Tablo 7.13'te görüldüğü gibi yazdırılmıştır.

Tablo 7.13 Birinci çalışma alanı derin öğrenme ile sınıflandırma doğruluk ölçütleri *(Ground, vegetation, building etiketleri sırasıyla zemin, bitki örtüsü ve bina sınıflarını temsil etmektedir. Precision, recall, f1-score, support, accuracy sırasıyla kesinlik, duyarlılık, f1-skor, nokta sayısını ve doğruluğu göstermektedir.)*

```
Training reached epoch's end. It took: 91.49027418295542 min.

              precision    recall  f1-score   support

      ground       0.85      0.92      0.89     70290
  vegetation       0.84      0.74      0.79     38308
    building       0.90      0.88      0.89     75002

    accuracy                           0.87    183600
   macro avg       0.86      0.85      0.85    183600
weighted avg       0.87      0.87      0.87    183600

0.8675925925925926
```

Tablo 7.12 ve Tablo 7.13'te hesaplanan doğruluk değerlerinin farklı olmasının sebebi, verilere yığınlama işlemi uygulanmış olmasıdır. Çalışma alanında 17066 tane test noktası seçilmiştir. Ancak yığınlama işlemi yapılırken test alanı için birbiriyle %50 kesişimli 612 adet yığın oluşturulmuş ve her yığın içerisine 300 nokta dahil edilmiştir. Bu sebeple test noktaları modele 183600 adet olarak verilmiştir ve doğruluk ölçütleri 18300 adet nokta üzerinden hesaplanmıştır. Yığınlar birbiriyle kesişimli olduğu için de tekrarlı noktalar içermektedir. Çıktı için tekrarlı noktaların tahminleri kıyaslanmış ve en çok hangi etiket atanmışsa tahmin sonucu o etiket olmuştur. Böylelikle nokta sayısı yine 17066'ya döndürülmüştür. İlk yapılan tabloda ise doğruluk 17066 nokta üzerinden hesaplanmıştır ve bu nihai doğruluktur.

4.5.6. İkinci Çalışma Alanı

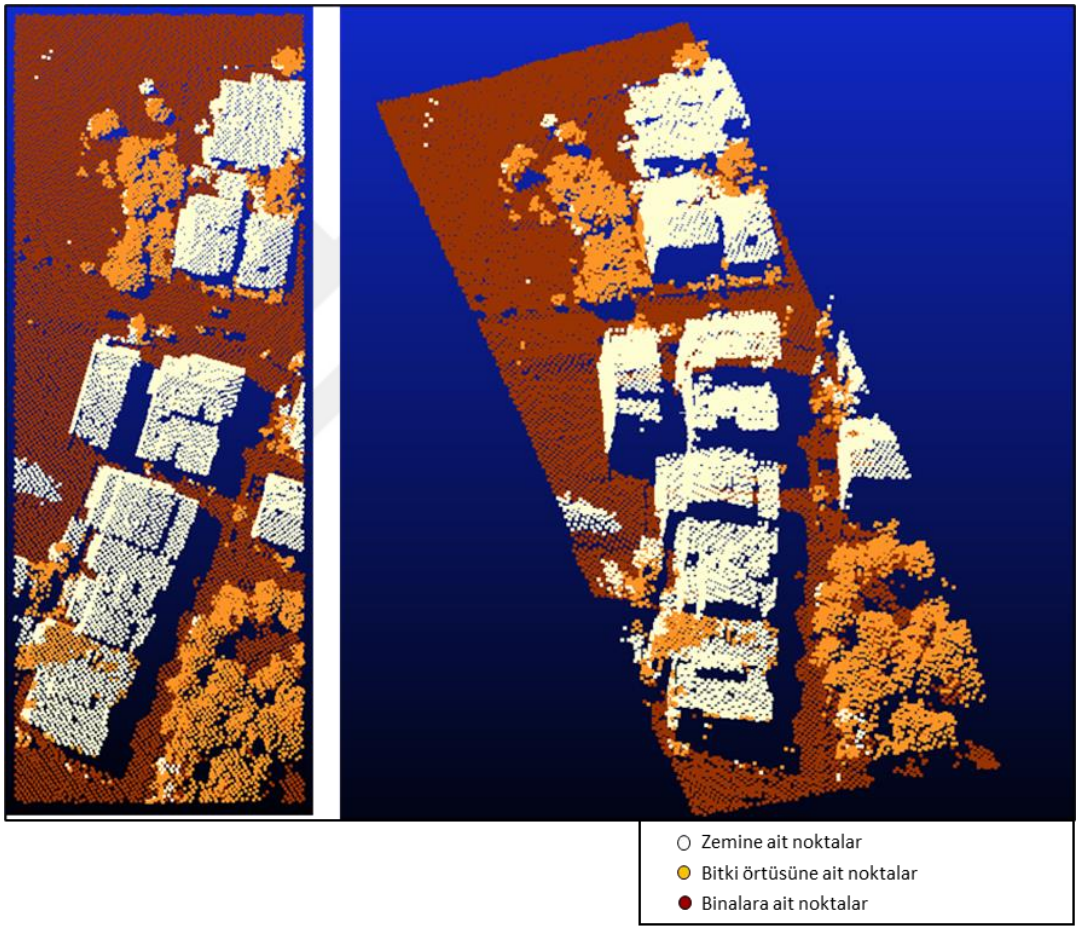
Ağın eğitimi yaklaşık 6 saat sürmüştür. Test alanının genel sınıflandırma doğruluğu %91.71 olarak hesaplanmıştır. Zemin noktaların çıkarımı %98.81, bitki örtüsü noktalarının çıkarımı %87.39, bina noktalarının çıkarımı %84.21 doğrulukla gerçekleşmiştir.

Şekil 7.17'de ikinci çalışma alanına ait test verisinin tahmin edilen sınıf etiketleriyle görselleştirilmesi verilmiştir. A görünümü nokta bulutunun tepeden, B görünümü

ise perspektifli görünümüdür. Tablo 7.14'te ise sınıflandırma hata matrisi verilmiştir.

Tablo 7.14 İkinci çalışma alanı Derin Öğrenme ile sınıflandırma hata tablosu

		Derin Öğrenme			
	Sınıflandırma	Bina	Bitki Örtüsü	Zemin	Doğruluk
Referans	Bina	**5993**	631	493	0,84
	Bitki Örtüsü	212	**3168**	245	0,87
	Zemin	95	21	**9604**	0,99
				Genel Doğruluk	**0,92**



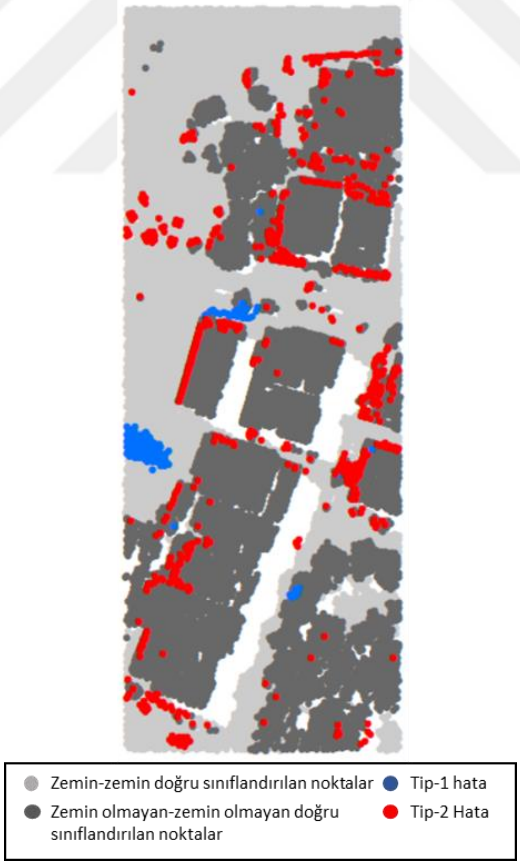
(A) (B)

Şekil 7.17 İkinci çalışma alanının derin öğrenme ile sınıflandırma sonucu (A)tepeden görünüm (B)perspektif görünüm

Tip-1 hatası %1.19, Tip-2 hatası %6.78, toplan hata ise %4.17 olarak hesaplanmıştır. Filtreleme doğruluğu %95.83'tür. Tablo 7.15 ile filtreleme hata matrisi, Şekil 7.18 ile de filtreleme sonucu hesaplanan Tip-1 ve Tip-2 hatalarının gösterimi verilmiştir.

Tablo 7.15 İkinci çalışma alanı Derin Öğrenme ile filtreleme hata tablosu

Derin Öğrenme					
	Filtreleme	Zemine Ait Noktalar	Zemine Ait Olmayan Noktalar		
Referans	Zemine Ait Noktalar	9604 (a)	116 (b)	Tip-1 Hatası	116 / (9604+116) = **%1.19**
	Zemine Ait Olmayan Noktalar	738 (c)	10004 (d)	Tip-2 Hatası	738 / (738+10004) = **%6.78**
				Toplam Hata	(116+738) / 20462 = **%4.17**



Şekil 7.18 İkinci alanın derin öğrenme ile filtrelenmesi sonucu Tip-1 (mavi) ve Tip-2 (kırmızı) hatalarının gösterimi

Sınıflandırma sonuçlarının kesinlik, duyarlılık ve f1-skor gibi doğruluk kriterleri hesaplanmıştır. Eğitimin tamamlanma süresiyle birlikte Tablo 7.16'da görüldüğü gibi yazdırılmıştır.

Tablo 7.16 İkinci çalışma alanı derin öğrenme ile sınıflandırma doğruluk ölçütleri

(Ground, vegetation, building etiketleri sırasıyla zemin, bitki örtüsü ve bina sınıflarını temsil etmektedir. Precision, recall, f1-score, support, accuracy sırasıyla kesinlik, duyarlılık, f1-skor, nokta sayısını ve doğruluğu göstermektedir.)

```
Training reached epoch's end. It took: 348.4152749737104 min.

              precision    recall  f1-score   support

      ground       0.93      0.98      0.95    300883
  vegetation       0.82      0.84      0.83    113631
    building       0.92      0.84      0.88    215486

    accuracy                           0.91    630000
   macro avg       0.89      0.89      0.89    630000
weighted avg       0.91      0.91      0.90    630000

0.9051603174603174
```

Tablo 7.15 ve Tablo 7.16'de hesaplanan doğruluk değerlerinin farklı olmasının sebebi, verilere yığınlama işlemi uygulanmış olmasıdır. Çalışma alanında 20462 tane test noktası seçilmiştir. Ancak yığınlama işlemi yapılırken test alanı için birbiriyle %50 kesişimli 2100 adet yığın oluşturulmuş ve her yığın içerisine 300 nokta dahil edilmiştir. Bu sebeple test noktaları modele 630000 adet olarak verilmiştir ve doğruluk ölçütleri 630000 adet nokta üzerinden hesaplanmıştır. Yığınlar birbiriyle kesişimli olduğu için de tekrarlı noktalar içermektedir. Çıktı için tekrarlı noktaların tahminleri kıyaslanmış ve en çok hangi etiket atanmışsa tahmin sonucu o etiket olmuştur. Böylelikle nokta sayısı yine 20462'ya döndürülmüştür. İlk yapılan tabloda ise doğruluk 20462 nokta üzerinden hesaplanmıştır ve bu nihai doğruluktur.

8
SONUÇ VE ÖNERİLER

İki çalışma alanında da noktaların tamamı KDH ile denetimsiz olarak sınıflandırılmıştır. Derin öğrenme ve Rastgele Orman algoritmasında ise verilerin yaklaşık %65i eğitim için kullanılmış ve %35'nin ise sınıf etiketleri tahmin edilmiştir.

EK-A'da görüldüğü üzere, çalışmada en iyi genel sınıflandırma ve filtreme sonuçları derin öğrenmeyle alınmıştır. Ancak sınıf bazlı doğruluklara bakıldığında, birinci çalışma için bitki örtüsü çıkarımında KDH'ın ve bina çıkarımında Rasgele Orman algoritmasının daha iyi sonuçlar verdiği görülmüştür. İkinci çalışma alanı içinse sınıflandırma bazlı doğruluklar için de en iyi sonuçların yine derin öğrenme ile alındığı görülmüştür. Fakat, en çok zaman alan, en çok parametre bilgisi gerektiren yöntem de derin öğrenme olmuştur. Yöntemlerin hızlarını ve karmaşıklığını göz önünde bulundurulursa, Rastgele Orman algoritmasının sonuçları da tatmin edicidir. Çünkü en az parametre karmaşıklığına ve 1 dakikadan az eğitim ve tahmin süresine Rastgele Orman yönteminde karşılaşılmıştır.

KDH'ın sınıf etiketlerine ihtiyaç duymadan iyi bir doğrulukla sınıflandırma ve filtreleme yapması bir avantaj iken, tüm kümeleme yöntemlerinde olduğu gibi oluşan kümelerin birleştirilmesi ve etiketlenmesi dezavantajıdır. Çok sayıda küme oluştuğunda, birleştirme işlemi karmaşık bir hale dönüşmektedir. Bu yüzden en az sayıda küme oluşturularak, en etkili nokta sınıflandırılmasının belirlenmesi amaçlanmış ve buna göre hata tolerans değeri belirlenmiştir. Bu çalışmada iki kez iç içe kümeleme işleminin gerçekleştirilmesi gerekmiştir.

Hata kaynakları genellikle kırsal alan problemlerinden kaynaklanmıştır. Saçaklar, farklı yapıların hem kendi içinde, hem de ağaçlarla birbirine girmiş durumları sebebiyle yanlış sınıflandırmalar görülmüştür.



Şekil 8.1 Hata kaynaklarına örnek sokak görüntüleri

Genel itibariyle bakıldığında bitki sınıflandırma doğruluğunun ayrık Lidar verisiyle çalışılmış olduğu göz önünde bulundurulunca sonuçlarının iyi seviyede olduğu söylenebilir, ancak tam dalga boyu formlu bir Lidar verisi bitki sınıflandırma doğruluğunu arttırabilir.

Özellikle zemin ve bina noktası çıkarımı olmak üzere genel olarak üç yöntemin de tatmin edici doğrulukta sonuçlar verdiği ve kimi durumlarda birbirlerini tamamladıkları görülmüştür. İleriki yıllarda bu yöntemlerin entegre biçimde kullanılarak sonuçların daha da iyileştirilmesi söz konusudur.

KAYNAKÇA

Adam, J. (2016). Machine Learning for Stratch. AI Sciences. ISBN-13: 978-1643160733

Axelsson, P. (1999). Processing of laser scanner data - Algorithms and applications. *ISPRS Journal of Photogrammetry and Remote Sensing*, *54*(2–3), 138–147. https://doi.org/10.1016/S0924-2716(99)00008-8

Axelsson, P. (2000). DEM Generation from Laser Scanner Data Using Adaptive TIN Models. *International Archives of Photogrammetry and Remote Sensing*, *33*(4), 110–117. https://www.isprs.org/proceedings/XXXIII/congress/part4/111_XXXIII-part4.pdf

Briese, C. (2010). Extraction of digital terrain models. *Airborne and Terrestrial Laser Scanning*, 147–150. https://vg.geoinfo.tuwien.ac.at/publications/np_airbornels_2001.pdf

Briese, C., & Pfeifer, N. (2001). Airborne laser scanning and derivation of digital terrain models. *Proceedings of the 5th Conference on Optical 3D Measurement Techniques*, *March*. https://vg.geoinfo.tuwien.ac.at/publications/np_airbornels_2001.pdf

Brovelli, M., Cannata, M., & Longoni, U. (2002). Managing and processing LIDAR data within GRASS. *Proceedings of the Open Source GIS - GRASS Users Conference*, *September*, 11–13. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Managing+and+processing+LIDAR+data+within+GRASS#0

Buczak, A. L., & Guven, E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys and Tutorials*, *18*(2), 1153–1176. https://doi.org/10.1109/COMST.2015.2494502

Buduma, N., & Locascio, N. (2017). Fundamentals of Deep Learning. O'Reilly Media, United States ıf America.

Camuñas-mesa, L. A., Linares-barranco, B., & Serrano-gotarredona, T. (2019). *Neuromorphic Spiking Neural Networks and Their Memristor-CMOS Hardware Implementations*. https://doi.org/10.3390/ma12172745

Chehata C., N., Guo, L., & Mallet, C. (2009). Airborne LiDAR feature selection for urban classification using random forests. *Geomatics and Information Science of Wuhan University*, 207–2012. https://doi.org/10.13203/j.whugis20130206

Chen, Q., Wang, H., Zhang, H., Sun, M., & Liu, X. (2016). A point cloud filtering approach to generating DTMs for steep mountainous areas and adjacent residential areas. *Remote Sensing*, *8*(1). https://doi.org/10.3390/rs8010071

Chen, W., Li, X., Wang, Y., Chen, G., & Liu, S. (2014). Forested landslide detection using LiDAR data and the random forest algorithm: A case study of the Three Gorges,

China. *Remote Sensing of Environment*, *152*(September), 291–301. https://doi.org/10.1016/j.rse.2014.07.004

Chen, Z., Gao, B., & Devereux, B. (2017). State-of-the-art: DTM generation using airborne LIDAR data. *Sensors (Switzerland)*, *17*(1). https://doi.org/10.3390/s17010150

Elmqvist, M., Jungert E., Lantz F., Persson A., S. U. (2001). Terrain Modelling and Analysis Using Laser Scanner Data. *International Archives of Photogrammetry and Remote Sensing*, *XXXIV*.

Elmqvist, M. (2002). Ground surface estimation from airborne laser scanner data using active shape models. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, *34*(3/A), 114–118.

Fu, Y. (2017). Combination of Random Forests and Neural Networks in Social Lending. *Journal of Financial Risk Management*, *06*(04), 418–426. https://doi.org/10.4236/jfrm.2017.64030

Gabriel, J. (2019). *Mastering Python Deep Learning*. AI Sciences. ISBN-13: 978-1-7335706-5-7

Giampouras, P., Charou, E., & Kesidis, A. (2013). Artificial neural network approach for land cover classification of fused Hyperspectral and LIDAR data. *IFIP Advances in Information and Communication Technology*, *412*(October), 255–261. https://doi.org/10.1007/978-3-642-41142-7_26

Girshick, R., Donahue, J., Darrell, T., Berkeley, U. C., & Malik, J. (2012). *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2–9.

Grebby, S., Naden, J., Cunningham, D., & Tansey, K. (2011). Integrating airborne multispectral imagery and airborne LiDAR data for enhanced lithological mapping in vegetated terrain. *Remote Sensing of Environment*, *115*(1), 214–226. https://doi.org/10.1016/j.rse.2010.08.019

Hu, X., & Yuan, Y. (2016). Deep-learning-based classification for DTM extraction from ALS point cloud. *Remote Sensing*, *8*(9). https://doi.org/10.3390/rs8090730

Huang, R., & Zhu, J. (2013). Using Random Forest to integrate lidar data and hyperspectral imagery for land cover classification. *International Geoscience and Remote Sensing Symposium (IGARSS)*, *January*, 3978–3981. https://doi.org/10.1109/IGARSS.2013.6723704

Jha, K., Doshi, A., & Patel, P. (2018). Intelligent irrigation system using artificial intelligence and machine learning: A Comprehensive Review. *ISSN : 2320-5407*. *September*. https://doi.org/10.21474/IJAR01/7959

Kang, X., Liu, J., & Lin, X. (2014). Streaming progressive TIN densification filter for airborne LiDAR point clouds using multi-core architectures. *Remote Sensing*, *6*(8), 7212–7232. https://doi.org/10.3390/rs6087212

Kayi, A., Erdoğan, M., & Eker, O. (2015). OPTECH HA-500 ve RIEGL LMS- Q1560 ile

Gerçekleştirilen LİDAR Test Sonuçları. *Harita Dergisi*, *153*(2), 42–46.

Kohonen, T., 1990. "The Self-Organizing Map." Proceedings of the IEEE 78 (9):1464-80. doi: 10.1109/5.58325

Kohonen T. 2001. Self Organizing Maps. 3rd ed. Berlin: Springer.

Kraus, K., & Pfeifer, N. (1998). Determination of terrain models in wooded areas with airborne laser scanner data. *ISPRS Journal of Photogrammetry and Remote Sensing*, *53*(4), 193–203. https://doi.org/10.1016/S0924-2716(98)00009-4

Kraus, K., & Pfeifer, N. (2001). Advanced Dtm Generation From Lidar Data. *International Archives of Photogrammetry and Remote Sensing*, *XXXIV*, 22–24.

Krose, B., & Smagt, P. (1996). An introduction to Neural Networks. (8th ed.). The University of Amsterdam.

Kwon, S. K., Jung, H. S., Baek, W. K., & Kim, D. (2017). Classification of forest vertical structure in South Korea from aerial orthophoto and lidar data using an artificial neural network. *Applied Sciences (Switzerland)*, *7*(10). https://doi.org/10.3390/app7101046

Li, L., Losser, T., Yorke, C., & Piltner, R. (2014). Fast inverse distance weighting-based spatiotemporal interpolation: A web-based application of interpolating daily fine particulate matter PM2:5in the contiguous U.S. using parallel programming and k-d tree. *International Journal of Environmental Research and Public Health*, *11*(9), 9101–9141. https://doi.org/10.3390/ijerph110909101

Liu, H., & Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences (Switzerland)*, *9*(20). https://doi.org/10.3390/app9204396

Ma, L., Zhou, M., & Li, C. (2017). Land covers classification based on Random Forest method using features from full-waveform LiDAR data. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *42*(2W7), 263–268. https://doi.org/10.5194/isprs-archives-XLII-2-W7-263-2017

Maturana, D., & Scherer, S. (2015). VoxNet: A 3D Convolutional Neural Network for real-time object recognition. *IEEE International Conference on Intelligent Robots and Systems*, *2015-Decem*, 922–928. https://doi.org/10.1109/IROS.2015.7353481

Mongus, D., & Žalik, B. (2014). Computationally Efficient Method for the Generation of a Digital Terrain Model From Airborne LiDAR. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *7*(1), 340–351.

Morris, J. T., Porter, D., Neet, M., Noble, P. A., Schmidt, L., Lapine, L. A., & Jensen, J. R. (2005). Integrating LIDAR elevation data, multi-spectral imagery and neural network modelling for marsh characterization. *International Journal of Remote Sensing*, *26*(23), 5221–5234. https://doi.org/10.1080/01431160500219018

Niemeyer, J., Rottensteiner, F., & Soergel, U. (2012). CONDITIONAL RANDOM FIELDS

for LIDAR POINT CLOUD CLASSIFICATION in COMPLEX URBAN AREAS. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *I–3*, 263–268. https://doi.org/10.5194/isprsannals-I-3-263-2012

Pfeifer, N. (2001). Derivation Of Digital Terrain Models In The Scop++ Environment. *OEEPE Workshop on Airborne Laserscanning and Interferometric SAR for Digital Elevation Models, ,* 13. http://www.lidar.com.br/arquivos/dem-scop-laser.pdf

Pfeifer, N., Reiter, T., Briese, C., & Rieger, W. (1999). Interpolation of high quality ground models from laser scanner data in forested areas. *International Archives of Photogrammetry and Remote Sensing*, *32*(3/W14), 31–36.

Pfeifer, N., Kostli, A., & Kraus, K. (1998). Interpolation and Filtering of Laser Scanner Data Implementation and First Results. *International Archives of Photogrammetry and Remote Sensing*, *33*, 153–159.

Pfeifer, N., Kostli, A., & Kraus, K. (1999). Restitution of Airborne Laser Scanner Data in Wooded Area. *GIS Geo-Information-Systems*, 12(2), 18–21.

Politz, F., & Sester, M. (2018). Exploring ALS and DIM data for semantic segmentation using CNNs. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *42*(1), 347–354. https://doi.org/10.5194/isprs-archives-XLII-1-347-2018

Politz, F., Kazimi, B., & Sester, M. (2018). Classification of Laser Scanning Data Using Deep Learning. *Remote Sensing*, *27*(9), 597–610. http://www.mdpi.com/2072-4292/8/9/730

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017-Janua*, 77–85. https://doi.org/10.1109/CVPR.2017.16

Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017). PointNet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems*, *2017-Decem*, 5100–5109.

Reitberger, J., Krzystek, P., & Stilla, U. (2008). Analysis of full waveform LIDAR data for the classification of deciduous and coniferous trees. *International Journal of Remote Sensing*, *29*(5), 1407–1431. https://doi.org/10.1080/01431160701736448

Roggero, M. (2001). Airborne Laser Scanning-Clustering in Raw Data. *International Archives of Photogrammetry and Remote Sensing*, *34*(3/W4), 227–232.

Sá, J. A. S., Almeida, A. C., Rocha, B. R. P., Mota, M. A. S., Souza, J. R. S., & Dentel, L. M. (2016). *Lightning Forecast Using Data Mining Techniques On Hourly Evolution Of The Convective Available Potential Energy*. *March*, 1–5. https://doi.org/10.21528/cbic2011-27.1

Salah, M., Trinder, J., Shaker, A., Hamed, M., & Elsagheer, A. (2009). Aerial Images and Lidar Data Fusion for Automatic Feature Extraction Using the Self-Organizing Map (Som) Classifier. *Scanning*, *XXXVIII*(2007), 317–322.

http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.157.2245

Sen, A., Suleymanoglu, B., & Soycan, M. (2020). Unsupervised extraction of urban features from airborne lidar data by using self-organizing maps. *Survey Review*, *52*(371), 150–158. https://doi.org/10.1080/00396265.2018.1532704

Sevgen, S. C., (2019). Airborne Lidar Data Classification in Complex Urban Area Using Random Forest: a Case Study of Bergama, Turkey. *International Journal of Engineering and Geosciences*, *1*, 45–51. https://doi.org/10.26833/ijeg.440828

Sithole, G. (2001). Filtering of Laser Altimetry Data Using a Slope Adaptive Filter. *Archives*, *XXXIV*, 22–24.

Sithole, George, & Vosselman, G. (2004). Experimental comparison of filter algorithms for bare-Earth extraction from airborne laser scanning point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, *59*(1–2), 85–101. https://doi.org/10.1016/j.isprsjprs.2004.05.004

Sohn, G., & Dowman, I. (2002). Terrain Surface Reconstruction By the Use of Tetrahedron Model With the Mdl Criterion. *International Archives of Photogrammetry and Remote Sensing*, *24*(3), 336–344.

Sotoudeh, H., Shafaat, O., Bernstock, J. D., Brooks, M. D., Elsayed, G. A., Chen, J. A., Szerip, P., Chagoya, G., Gessler, F., Sotoudeh, E., Shafaat, A., & Friedman, G. K. (2019). Artificial intelligence in the management of glioma: Era of personalized medicine. *Frontiers in Oncology*, *9*(AUG). https://doi.org/10.3389/fonc.2019.00768

Şen, A. (2013) Kartografik Genelleştirmede Seçme/Eleme İşlemleri için Yapay Zeka Yöntemlerinin Akarsu Ağına Uygulanabilirliği. Doctorate thesis, Yıldız Tecnical Üniversity

Takma, Ç., & Atıl, H., (2006). Bootstrap Metodu ve Uygulanışı Üzerine Bir Çalışma 2. Güven Aralıkları, Hipotez Testi ve Regresyon Analizinde Bootstrap Metodu. *Ege Üniversitesi Ziraat Fakültesi Dergisi*, *43*(2), 63–72. https://doi.org/10.20289/eüzfd.61671

Wack R., W. A. (2002). DIGITAL TERRAIN MODELS FROM AIRBORNE LASER SCANNER DATA - A GRID BASED APPROACH. *International Archives of Photogrammetry and Remote Sensing*, *XXXIV*(3B), 293–296.

Wang, C., Shu, Q., Wang, X., Guo, B., Liu, P., & Li, Q. (2019). A random forest classifier based on pixel comparison features for urban LiDAR data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 75–86. https://doi.org/10.1016/j.isprsjprs.2018.12.009

Wang, G., Pu, P., & Shen, T. (2020). An efficient gene bigdata analysis using machine learning algorithms. *Multimedia Tools and Applications*, *79*(15–16), 9847–9870. https://doi.org/10.1007/s11042-019-08358-7

Wang, L., & Fu, K. (2009). Artificial Neural Networks. Wiley Encyclopedia of Computer Science and Engineering. 181-188. Online ISBN: 9780470050118,

doi: 10.1002/9780470050118

Winiwarter, L., Mandlburger, G., Schmohl, S., & Pfeifer, N. (2019). Classification of ALS Point Clouds Using End-to-End Deep Learning. *PFG - Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, *87*(3), 75–90. https://doi.org/10.1007/s41064-019-00073-0

Xue, H., Sun, S., Venkataramani, G., & Lan, T. (2019). Machine Learning-Based Analysis of Program Binaries: A Comprehensive Study. *IEEE Access*, *7*(June), 65889–65912. https://doi.org/10.1109/ACCESS.2019.2917668

Yan, J., and J. Thill. 2008. Visual Exploration of Spatial Interaction Data with Self-Organizing Maps. Chap. 4 In: Agarwal P, Skupin A, editors. Self-Organizing Maps: Applications in Geographic Information Science England: Willey; p. 67-85

Zaletnyik, P., Laky, S., & Toth, C. (2010). LIDAR waveform classification using self-organizing map. *American Society for Photogrammetry and Remote Sensing Annual Conference 2010: Opportunities for Emerging Geospatial Technologies*, *2*(1999), 1055–1066.

Zhang, J., Lin, X., & Ning, X. (2013). SVM-Based classification of segmented airborne LiDAR point clouds in urban areas. *Remote Sensing*, *5*(8), 3749–3775. https://doi.org/10.3390/rs5083749

Zhou, Z., & Gong, J. (2018). Automated residential building detection from airborne LiDAR data with deep neural networks. *Advanced Engineering Informatics*, *36*(February), 229–241. https://doi.org/10.1016/j.aei.2018.04.002

URL1: “Airborne laser scanning”. Accessed on: May 15, 2021. [Online]. Available on: http://gmv.cast.uark.edu/scanning2/airborne-laser-scanning/

URL2: Arcgis, “PointCloudReturnFilter”. Accessed on: May 15, 2021. [Online]. Available on: https://developers.arcgis.com/javascript/latest/api-reference/esri-layers-pointCloudFilters-PointCloudReturnFilter.html

URL3: "Sigmoid, Softmax and their deriatives". Accessed on : May 15, 2021. [Online]. Available on: https://themaverickmeerkat.com/2019-10-23-Softmax/

URL4: "sklearn.ensemble.RandomForestClassifier" Accessed on: May 1,2021. [Online]. Available on: https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

URL5: "SciPy.org" Accessed on: May 1,2021. [Online]. Available on: https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.cKDTree.query_ball_tree.html#scipy.spatial.cKDTree.query_ball_tree

A

ÜÇ YÖNTEMİN KARŞILAŞTIRILDIĞI TABLO

Yöntemler	Birinci Çalışma Alanı			İkinci Çalışma Alanı		
	Nokta sayısı	Sınıflandırma Doğruluğu	Filtreleme Doğruluğu	Nokta sayısı	Sınıflandırma Doğruluğu	Filtreleme doğruluğu
Kendini Düzenleyen Haritalar	Toplam: 46438	Zemin: %88.46 Bitki örtüsü: %81.36 Bina: %73.33 Genel: **%80.99**	**%83.59** *Tip-1 hata: %11.54* *Tip-2 hata: %19.43*	Toplam: 58725	Zemin: %94.90 Bitki örtüsü: %54.49 Bina: %79.79 Genel: **%80.97**	**%86.98** *Tip-1 hata: %5.10* *Tip-2 hata: %18.74*
Rastgele Orman Algoritması	Eğitim: 29372 Test: 17066 Toplam: 46438	Zemin: % 88.26 Bitki örtüsü: %66.54 Bina: %90.89 Genel: **%84.27**	**%90.61** *Tip-1 hata: %11.74* *Tip-2 hata: %8.00*	Eğitim: 38263 Test: 20462 Toplam: 58725	Zemin: %86.23 Bitki örtüsü: %75.56 Bina: %75.40 Genel: **%80.57**	**%86.93** *Tip-1 hata: %13.77* *Tip-2 hata: %12.44*
Derin Öğrenme	Eğitim: 29372 Test: 17066 Toplam: 46438	Zemin: %94.90 Bitki örtüsü: %75.77 Bina: %88.58 Genel: **%87.96**	**%91.79** *Tip-1 hata: %6.04* *Tip-2 hata: %10.05*	Eğitim: 38263 Test: 20462 Toplam: 58725	Zemin: %98.81 Bitki örtüsü: %87.39 Bina: %84.21 Genel: **%91.71**	**%95.83** *Tip-1 hata: %1.19* *Tip-2 hata: %6.78*

TEZDEN ÜRETİLMİŞ YAYINLAR

Konferans Bildirileri

B. Bayaslı ve A. Sen, "Hava Lidar verilerinin yapay sinir ağları kullanılarak sınıflandırılması," *17. Harita Bilimsel Teknik Kurultayı,* Ankara, 25-27 Nisan 2019.

Makaleler

A. Şen and B. Bayaslı, "Hava Lidar verilerinin denetimsiz yapay sinir ağları kullanılarak filtrelenmesi," *Jeodezi ve Jeoinformasyon Dergisi*, cilt. 8, s. 1, ss. 18–29, 2021, doi: 10.9733/jgg.2021r0002.t.