

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK- ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

LİDAR NOKTA BULUTLARININ DERİN ÖĞRENME VE MAKİNE
ÖĞRENMESİ YÖNTEMLERİ İLE SINIFLANDIRILMASI

YÜKSEK LİSANS TEZİ

Nisa KOBAN

EKİM 2025
TRABZON



KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK- ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

**LİDAR NOKTA BULUTLARININ DERİN ÖĞRENME VE MAKİNE
ÖĞRENMESİ YÖNTEMLERİ İLE SINIFLANDIRILMASI**

Nisa KOBAN

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünde
"ELEKTRONİK YÜKSEK MÜHENDİSİ"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 15 / 09 / 2025

Tezin Savunma Tarihi : 27 / 10 / 2025

Tez Danışmanı : Dr. Öğretim Üyesi Emrah BENLİ

Trabzon 2025

**KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**Elektrik-Elektronik Mühendisliği Anabilim Dalında
Nisa KOBAN Tarafından Hazırlanan**

**LİDAR NOKTA BULUTLARININ DERİN ÖĞRENME VE MAKİNE
ÖĞRENMESİ YÖNTEMLERİ İLE SINIFLANDIRILMASI**

**başlıklı bu çalışma, Enstitü Yönetim Kurulunun 02 / 10 / 2025 gün ve 2122
sayılı kararıyla oluşturulan jüri tarafından yapılan sınavda
YÜKSEK LİSANS TEZİ
olarak kabul edilmiştir.**

Jüri Üyeleri

Başkan : Doç. Dr. Selahattin KOŞUNALP
Üye : Dr. Öğr. Üyesi Oğuzhan ÇAKIR
Üye : Dr. Öğr. Üyesi Emrah BENLİ

**Prof. Dr. Asim KADIOĞLU
Enstitü Müdürü**

ÖNSÖZ

‘Lidar Nokta Bulutlarının Derin Öğrenme ve Makine Öğrenmesi Yöntemleri ile Sınıflandırılması’ adlı bu çalışma Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektrik – Elektronik Mühendisliği Anabilim Dalı Yüksek Lisans Programı’nda hazırlanmıştır.

Çalışmalarım boyunca fikir ve önerileriyle beni yönlendiren, tezimin gelişmesine katkıda bulunan, her aşamada bilgi ve tecrübelerinden faydalanmamı sağlayan, değerli yorumlarıyla motivasyonumu artırıp bana yol gösteren danışman hocam Sayın Dr. Öğr. Üyesi Emrah BENLİ’ye sonsuz teşekkürlerimi sunarım.

Son olarak, yüksek lisans tez çalışmam süresince beni her koşulda destekleyen, sevgileriyle bana güç veren aileme en içten teşekkürlerimi sunarım.

Nisa KOBAN
Trabzon 2025

TEZ ETİK BEYANNAMESİ

Yüksek Lisans Tezi olarak sunduğum “ Lidar Nokta Bulutlarının Derin Öğrenme ve Makine Öğrenmesi Yöntemleri ile Sınıflandırılması” başlıklı bu çalışmayı baştan sona kadar danışmanım Dr. Öğr. Üyesi Emrah BENLİ‘ nin sorumluluğunda tamamladığımı, verileri/örnekleri kendim topladığımı, deneyleri/analizleri ilgili laboratuvarlarda yaptığımı/yaptırdığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynakçada eksiksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim. 27/10/2025

Nisa KOBAN

İÇİNDEKİLER

	<u>Sayfa No</u>
ÖNSÖZ.....	III
TEZ ETİK BEYANNAMESİ.....	IV
İÇİNDEKİLER.....	V
ÖZET	VIII
SUMMARY	IX
ŞEKİLLER DİZİNİ	X
TABLolar DİZİNİ.....	XII
SEMBOLLER VE KISALTMALAR DİZİNİ	XIII
1. GENEL BİLGİLER	1
1.1. Giriş.....	1
1.2. Tezin Amacı	2
1.3. Lidar Yapısal Bileşenleri.....	4
1.4. Nokta Bulutu Verileri.....	5
1.4.1. Sınırlayıcı Kutu Yapısı.....	6
1.5. Sınıflandırma Performans Ölçütleri	10
1.5.1. Doğruluk	10
1.5.2. Kesinlik (Precision).....	10
1.5.3. Duyarlılık/Gerçek Pozitif Oranı	10
1.5.4. F1 Skoru	11
1.5.5. Karışıklık Matrisi (Confusion Matrix).....	11
2. YAPILAN ÇALIŞMALAR.....	12
2.1. Veri Kümeleri ve Etiketleme Yapısı	12
2.2. Veri Ön işleme Adımları	12
2.2.1. Etiketli Noktaların Elde Edilmesi	13
2.2.2. Sınıf Filtreleme ve Hedef Seçimi.....	13
2.2.3. Sınıf Denge Analizi ve Veri Normalizasyonu	16
2.2.4. Veri Formatı ve Dönüşümü.....	16
2.3. Makine Öğrenmesi Yaklaşımları.....	19
2.3.1. Random Forest (Rastgele Orman) Algoritması	19
2.3.2. Multi-Layer Perceptron (Çok Katmanlı Algılayıcı) (ML-MLP)	20

2.3.3.	K-En Yakın Komşu (K-Nearest Neighbors, KNN) Yöntemi	22
2.4.	Derin Öğrenme Yaklaşımları	23
2.4.1.	Derin Öğrenme MLP (DL-MLP) Yöntemi	24
2.4.2.	Convolutional Neural Network (CNN).....	25
2.4.3.	Pointnet-Mini	27
2.5.	Deney Ortamı	29
2.6.	Kullanılan Kütüphane, Fonksiyon ve Modüller	30
2.6.1.	Numpy Kütüphanesi	30
2.6.2.	Google.colab Modülü.....	31
2.6.3.	Matplotlib Kütüphanesi.....	32
2.6.4.	Ploty Kütüphanesi	32
2.6.5.	Scikit-learn (sklearn) Kütüphanesi.....	33
2.6.6.	PyTorch (torch) Kütüphanesi.....	34
2.6.7.	KNeighborsClassifier.....	36
2.6.8.	Yardımcı Kütüphaneler (os, json).....	36
2.7.	Deney Kurulumu	38
2.8.	Temel Modeller ve Deney Süreçleri	39
2.8.1.	Random Forest Yöntemi ile Deneysel Sonuçlar	39
2.8.2.	DL-MLP Yöntemi ile Deneysel Sonuçlar.....	48
2.8.3.	ML-MLP Yöntemi ile Deneysel Sonuçlar	52
2.8.4.	KNN Yöntemi ile Deneysel Sonuçlar	55
2.8.5.	PointNet – Mini Yöntemi ile Deneysel Sonuçlar.....	59
2.8.6.	CNN Yöntemi ile Deneysel Sonuçlar	63
2.9.	Ek Denemeler ve Deneylere Dahil Edilmeyen Çalışmalar	69
2.9.1.	Lidar Nokta Bulutu ve Kamera Görüntüleri Eşleştirme Denemeleri	69
2.9.2.	Extra Tree (Karar Ağaçları) Yöntemi ile Deneme.....	72
3.	BULGULAR VE TARTIŞMA	76
3.1.	Deney Düzeni	76
3.2.	Genel Performans Karşılaştırması.....	77
3.2.1.	Araba Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması	80
3.2.2.	Yaya Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması.....	81
3.2.3.	Otobüs Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması	83

3.2.4. Kamyon Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması	85
3.2.5. Motosiklet Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması	87
3.2.6. Bariyer Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması	88
3.2.7. Bisiklet Rafı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması	90
3.3. Genel Sonuçlar ve Karşılaştırma	92
4. SONUÇLAR VE ÖNERİLER	94
5. KAYNAKLAR	95
ÖZGEÇMİŞ	



Yüksek Lisans Tezi

ÖZET

LİDAR NOKTA BULUTLARININ DERİN ÖĞRENME VE MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE SINIFLANDIRILMASI

Nisa KOBAN

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı
Danışman: Dr. Öğr. Üyesi Emrah BENLİ
2025, 100 Sayfa

Düzensiz ve yoğun yapıya sahip nokta bulutlarının iç mekân ve dış mekân uygulamalarında doğru şekilde işlenmesi, bina modelleme, navigasyon ve otonom sistemler için önemli bir gereklilik olarak öne çıkmaktadır. Bu çalışmada Lidar nokta bulutu verilerinin farklı makine öğrenmesi ve derin öğrenme yöntemleri kullanılarak sınıflandırılması ve bu yöntemlerin performanslarının karşılaştırılması amaçlanmıştır.

Çalışmada NuScenes veri setinin Lidarseg-mini alt kümesi kullanılmış ve nokta bulutları yedi farklı sınıfa ayrılmıştır. Deneysel çalışmalarda hem klasik makine öğrenmesi algoritmaları Random Forest, K-En Yakın Komşu, ML-MLP hem de derin öğrenme tabanlı yöntemler DL-MLP, CNN, PointNet Mini uygulanmış, modellerin doğruluk, F1, duyarlılık ve kesinlik metrikleri üzerinden karşılaştırmaları yapılmıştır.

Elde edilen sonuçlara göre, Random Forest gibi klasik yöntemler hızlı düşük maliyetli çözümlerle daha yüksek doğruluk ve genelleme kabiliyeti sunarken, derin öğrenmede CNN ve PointNet tabanlı yaklaşımlar daha düşük doğruluk ve genelleme kabiliyeti göstermiştir. Bununla birlikte, sınıf dengesizliği sorunu küçük örneklerle sahip kategorilerde performans düşüşüne yol açmıştır.

Sonuçlar, makine öğrenmesi yöntemlerinden Random Forest algoritmasının en yüksek genel doğruluk ve tutarlı sonuçları verdiğini göstermiştir. Derin öğrenme yöntemleri arasında ise PointNet mini mimarisi özellikle nokta tabanlı özellik çıkarımı sayesinde güçlü bir performans sergilemiştir.

Anahtar Kelimeler: Derin Öğrenme, Nokta Bulutları, Makine Öğrenmesi, Sınıflandırma

PhD. Thesis

SUMMARY

CLASSIFICATION OF LIDAR POINT CLOUDS USING DEEP LEARNING AND MACHINE LEARNING METHODS

Nisa KOBAN

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Electrical and Electronics Engineering Graduate Program
Supervisor: Asst. Prof. Emrah BENLİ
2025, 100 Pages

The accurate processing of point clouds with irregular and dense structures in indoor and outdoor applications is emerging as an important requirement for building modeling, navigation, and autonomous systems. This study aims to classify Lidar point cloud data using different machine learning and deep learning methods and to compare the performance of these methods.

The Lidarseg-mini subset of the NuScenes dataset was used in the study, and the point clouds were divided into seven different classes. In the experimental studies, both classical machine learning algorithms (Random Forest, K-Nearest Neighbor, ML-MLP) and deep learning-based methods (DL-MLP, CNN, PointNet Mini) were applied, and the models were compared based on accuracy, F1, sensitivity, and precision metrics.

According to the results obtained, classical methods such as Random Forest offer higher accuracy and generalization ability with fast, low-cost solutions, while CNN and PointNet-based approaches in deep learning showed lower accuracy and generalization ability. However, the class imbalance problem led to a decrease in performance in categories with small samples.

The results showed that the Random Forest algorithm among machine learning methods provided the highest overall accuracy and consistent results. Among deep learning methods, the PointNet mini architecture demonstrated strong performance, particularly due to point-based feature extraction.

Key Words: Classification, Deep Learning, Machine Learning, Point Clouds

ŞEKİLLER DİZİNİ

	<u>Sayfa No</u>
Şekil 1. 7 Sınıfa ait ilk on bin nokta.....	6
Şekil 2. Kutu içindeki ham noktalar	8
Şekil 3. Kutu içinde renkli etiketlenmiş sınıflar	9
Şekil 4. Lidar noktalarının işleme süreci	13
Şekil 5. Sınıf başına nokta sayılarının histogramı	15
Şekil 6. CNN mimarisi.....	26
Şekil 7. Pointnet-Mini katman yapısı	28
Şekil 8. Numpy kütüphanesi örnek kod çıktısı	31
Şekil 9. Google.colab modülü örnek kod çıktısı.....	31
Şekil 10. Matplotlib kütüphanesi örnek kod çıktısı	32
Şekil 11. Plotly kütüphanesi örnek kod çıktısı.....	33
Şekil 12. Plotly kütüphanesi kullanım örneği	33
Şekil 13. Sklearn kütüphanesi örnek kod çıktısı.....	34
Şekil 14. PyTorch kütüphanesi örnek kod çıktısı	35
Şekil 15. KNeighborsClassifier örnek kod çıktısı.....	36
Şekil 16. KNeighborsClassifier kullanım kod örneği	36
Şekil 17. Os yardımcı kütüphane örnek kod çıktısı	37
Şekil 18. Tartfile yardımcı kütüphane örnek kod çıktısı.....	37
Şekil 19. 2 sınıf için Random Forest sınıf bazlı başarı raporu.....	39
Şekil 20. Sınıfta sınıflandırma sonucunun karışıklık matrisi.....	40
Şekil 21. 2 sınıflı sınıflandırmada doğru-hatalı dağılım	41
Şekil 22. 3 sınıf için sınıf bazlı başarı raporu	42
Şekil 23. 3 sınıfta sınıflandırma sonucunun karışıklık matrisi	42
Şekil 24. 3 sınıfta sınıf bazlı metrik raporu.....	43
Şekil 25. 7 sınıf için sınıf bazlı başarı raporu	44
Şekil 26. 7 sınıf için karışıklık matrisi	45
Şekil 27. 7 sınıf için normalize edilmiş karışıklık matrisi	46
Şekil 28. Sınıf bazlı IoU başarı raporu.....	47
Şekil 29. DL-MLP sınıflandırma bazlı doğruluk raporu	48
Şekil 30. 7 sınıf için normalize karışıklık matrisi	49

Şekil 31.	DP-MLP'nin eğitim/doğrulama kaybı ve doğruluk eğrileri	50
Şekil 32.	ML-MLP 7 sınıf için performans raporu	52
Şekil 33.	7 sınıf için ML-MLP karışıklık matrisi	53
Şekil 34.	ML-MLP eğitim ve doğrulama doğruluğu grafiği.....	53
Şekil 35.	100 iterasyon ile bazlı performans raporu	54
Şekil 36.	K = 3 için sınıf bazlı doğruluk raporu	55
Şekil 37.	K = 11 için sınıf bazlı doğruluk raporu	56
Şekil 38.	K = 7 için sınıf bazlı doğruluk raporu	56
Şekil 39.	Sınıf bazlı IoU değeri.....	57
Şekil 40.	K = 3 için karışıklık matrisi	58
Şekil 41.	K= 3 için sınıf bazlı kesinlik/ duyarlılık/ F1 skor.....	59
Şekil 42.	PointNet-Mini sınıf bazlı doğruluk raporu	60
Şekil 43.	PointNet-Mini 7 sınıf için karışıklık matrisi.....	61
Şekil 44.	7 sınıf için normalize edilmiş karışıklık matrisi	61
Şekil 45.	PointNet-Mini Kayıp Grafiği.....	62
Şekil 46.	PointNet Mini doğruluk grafiği	63
Şekil 47.	CNN sınıf bazlı başarı raporu	64
Şekil 48.	CNN 7 sınıf için karışıklık matrisi	64
Şekil 49.	Cnn her sınıf için skor değerleri	65
Şekil 50.	Sınıf bazlı IoU değeri.....	67
Şekil 51.	Cnn modeli eğitim ve doğrulama grafiği.....	68
Şekil 52.	Yaya sınıfına ait noktaların ön kamera görüntüleri ile eşleşmesi.....	70
Şekil 53.	Arka sol kamera görüntüsünde yaya noktasının eşleşmesi.....	71
Şekil 54.	Karanlık ortamda Lidar noktasının kamera görüntüsüne eşleşmesi	71
Şekil 55.	Özellik önemleri grafiği.....	72
Şekil 56.	Doğru ve yanlış sınıflandırılan noktalar	73
Şekil 57.	Tahmin edilen sınıflara göre 3B nokta bulutları.....	74
Şekil 58.	Sınıf bazlı doğruluk grafiği.....	74

TABLÖLAR DİZİNİ

	<u>Sayfa No</u>
Tablo 1. Sınıf etiketlerinin renk eşleřtirmesi	8
Tablo 2. Hedeflenen 7 sınıf.....	14
Tablo 3. Veri Formatları	18
Tablo 4. Uygulanan yöntemlerin genel doğruluk performansı	78
Tablo 5. Araba sınıfı performans sonuçları	80
Tablo 6. Yaya sınıfı performans sonuçları.....	82
Tablo 7. Otobüs sınıfı performans sonuçları.....	84
Tablo 8. Kamyon sınıfı performans sonuçları	85
Tablo 9. Motosiklet sınıfı performans sonuçları.....	87
Tablo 10. Bariyer sınıfı performans sonuçları	89
Tablo 11. Bisiklet rafı sınıfı performans sonuçları	91

SEMBOLLER VE KISALTMALAR DİZİNİ

CNN	: Convolutional Neural Network (Evrişimli Sinir Ağı)
RO	: Rastgele Orman
ÇKA	: Çok Katmanlı Algılayıcı
KDH	: Kendini Düzenleyen Haritalar
IoU	: Intersection Over Union (Kesişim/Birleşim Oranı)
MIoU	: Mean Intersection Over Union (Ortalama Kesişim/Birleşim Oranı)
KNN	: K-Nearest Neighbors (K-En Yakın Komşu Algoritması)
MLP	: Multi Layer Perceptron (Çok Katmanlı Algılayıcı)
ML-MLP	: Machine Learning Multi Layer Perceptron (Makine Öğrenmesi Çok Katmanlı Algılayıcı)
DL-MLP	: Deep Multi Layer Perceptron (Derin Çok Katmanlı Algılayıcı)
ReLU	: Rectified Linear Unit (Düzeltilmiş Lineer Fonksiyon)
2B	: İki Boyut
3B	: Üç Boyut
RF	: Random Forest (Rastgele Orman)
GPU	: Graphics Processing Unit (Grafik İşlem Birimi)
CPU	: Central Processing Unit (Merkezi İşlem Birimi)
TP	: True Positive (Pozitif Doğru)
TN	: True Negative (Negatif Doğru)
FP	: False Positive (Yanlış Pozitif)
FN	: False Negative (Negatif Yanlış)

1. GENEL BİLGİLER

1.1. Giriş

Son yıllarda dijital sistemlerdeki hızlı gelişmeler, çevresel verilerin toplanması ve işlenme biçimlerinde önemli dönüşümlere neden olmuştur. Bu teknolojiler arasında Lidar (Light Detection and Ranging) sistemleri, yüksek hassasiyetli üç boyutlu uzamsal verileri ile öne çıkmaktadır. Lazer ışınları göndererek cisimlerden yansıyan sinyalleri ölçen bu sistemler, karmaşık ortamları ayrıntılı biçimde temsil eden nokta bulutu verileri oluşturur. Elde edilen bu nokta bulutları, otonom sürüşten şehir planlamasına, robotik algılamadan haritalamaya kadar pek çok alanda temel veri kaynağı haline gelmiştir. Lidar verileri genellikle milyonlarca üç boyutlu noktadan oluşur ve her nokta x , y , z konum bilgisine ek olarak yoğunluk gibi çeşitli özellikler taşır. Bu verilerden anlamlı bilgi elde edilebilmesi için yapay zekâ destekli yöntemlerin, özellikle de makine öğrenmesi ve derin öğrenme tekniklerinin devreye girmesi kaçınılmaz hale gelmektedir. Sınıflandırma işlemi, nokta bulutu verilerinin semantik anlam kazandırılarak, her noktanın ait olduğu nesne türüne göre etiketlenmesini sağlamaktadır. Bu işlem, ortamın detaylı analizine olanak tanırken, karar verme süreçlerinin de temelini oluşturur.

Bu tez çalışmasında, Lidar verilerinin sınıflandırılmasına yönelik derin öğrenme ve makine öğrenmesi yaklaşımları kullanılmıştır. Çalışmada, gerçek bir Lidar veri kümesi olan nuScenes'in Lidarseg-mini sürümü kullanılarak 7 farklı sınıf için sınıflandırma işlemi gerçekleştirilmiştir. Kullanılan sınıflar arasında araçlar, yayalar, motosikletler, bariyerler ve sabit nesne yapıları gibi farklı kategoriler yer almaktadır. Veri seti, Velodyne HDL-32E Lidar sensörü ile toplanmış olup, 360° görüş alanına sahip, yüksek yoğunluklu taramalar içermektedir. Çalışmanın tüm aşamaları Google Colab platformunda yürütülmüş olup veriler burada işlenmiş ve görselleştirilmiştir. Bu platformun GPU desteği sayesinde eğitim süreleri önemli ölçüde kısaltılmıştır. Ek olarak Python, NumPy, Pandas ve Plotly gibi kütüphaneler aracılığıyla grafiksel analiz ve etkileşimli görselleştirmeler gerçekleştirilmiştir.

Yöntemsel olarak çalışmada hem klasik makine öğrenmesi algoritmaları hem de derin öğrenme mimarileri kullanılmıştır. Çok Katmanlı Perceptron (MLP) ve 1D Convolutional Neural Network (1D CNN), Random Forest, , KNN, PointNet-Mini gibi

farklı yöntemler ile Lidar verisi üzerinde deneyler gerçekleştirilmiş, sınıflandırma doğruluğu ve sınıf bazlı başarımlar ölçütleri analiz edilmiştir. Ayrıca, sınırlı başarı gösteren Extra Tree, görüntü ve nokta eşleştirme gibi denemeler değerlendirilmiş ve performans açısından neden yetersiz kaldığı tartışılmıştır. Sonuç olarak, bu tez çalışmasında Lidar verilerinin nokta bulutu sınıflandırılması konusunda, farklı öğrenme modelleri karşılaştırılarak algoritmaların performanslarını ortaya koymayı amaçlanmıştır. Sınıf bazlı doğruluk, karışıklık matrisleri ve F1 skorları gibi değerlendirme metrikleri yardımıyla modellerin güçlü ve zayıf yönleri incelenmiştir.

1.2. Tezin Amacı

Lidar verisi yoğunlukla milyonlarca nokta içerdiğinden, bu noktaların sınıflandırılması sabit nesnelerin doğru şekilde tanımlanmasında kritik bir rol oynamaktadır. Sınıflandırma süreci, her bir noktanın ait olduğu sınıfın belirlenmesi anlamına gelir. Son yıllarda, nokta bulutlarının daha etkin kullanılabilmesi için özellik tabanlı sınıflandırma yöntemlerine yönelik araştırmalar artmıştır. Yoğunluk, yükseklik veya eğim gibi öznitelikler, sınıflandırma algoritmalarına ek bilgi sağlayarak karar verme sürecini kolaylaştırmaktadır. Liu ve Yan (2021), Lidar verilerinden elde edilen bu tür özniteliklerin 3B iç mekân modellemesi ve sınıflandırma çalışmalarında önemli katkılar sağladığını göstermiştir (Liu & Yan, 2021). Benzer şekilde Pirotti ve arkadaşları (2020), özellikle iç mekân uygulamalarında özellik çıkarımının sınıflandırma başarısını artırdığını ve analiz doğruluğuna önemli katkılar sunduğunu vurgulamıştır (Pirotti vd., 2020).

Son dönemde yapay zekâ ve makine öğrenmesi teknikleri de bu alanda yoğun şekilde kullanılmaktadır. Zhang ve arkadaşları (2016), derin öğrenme algoritmalarının Lidar verisinin doğru sınıflandırılmasında büyük potansiyel sunduğunu belirtmiştir (Zhang vd., 2016). Atik ve arkadaşları (2022) ise insansız hava aracı görüntülerinden elde ettikleri nokta bulutu verisini dört sınıfa ayırmış ve Rastgele Orman (RO) ile Çok Katmanlı Algılayıcı (ÇKA) yöntemlerini kıyaslamıştır. Çalışmalarında RO algoritması ile %78,54 ÇKA ile %89,88 genel doğruluk elde edilmiş olup, bu durum derin öğrenme tabanlı yaklaşımların klasik yöntemlere kıyasla daha yüksek doğruluk sağladığını göstermektedir (Atik vd., 2022).

Nokta bulutlarının sınıflandırılmasına yönelik derin öğrenme tabanlı çalışmalar da hız kazanmıştır. Minemura ve arkadaşları (2018), yalnızca nokta bulutu verisi kullanarak

kentsel alanlarda nesne tespiti yapabilen optimize edilmiş bir evrişimli sinir ağı geliştirmiştir. Genişletilmiş evrişim yapıları sayesinde ağ, algı alanını artırmış ve işlem süresini yaklaşık %30 kısaltmıştır. Ayrıca 50 FPS hızında çalışabilmesi, yöntemin gerçek zamanlı uygulamalara uygunluğunu ortaya koymuştur (Minemura vd., 2018). Benzer şekilde, Şen ve Bayaslı (2021) Lidar verilerini denetimsiz yapay sinir ağları ile filtreleyerek Kendini Düzenleyen Haritalar (KDH) tekniğini kullanmış ve kümeleri uydu görüntüleriyle doğrulamıştır. Bu yöntem, denetimsiz tekniklerin de Lidar sınıflandırmasında etkin olabileceğini ortaya koymaktadır (Şen & Bayaslı, 2021).

Derin öğrenme alanında öne çıkan çalışmalardan biri Qi ve arkadaşlarının (2017) geliştirdiği PointNet++'tır. Bu yöntem, nokta bulutlarının mekânsal yapısını hiyerarşik bir ağ üzerinden öğrenmekte ve farklı seviyelerdeki özellikleri yakalayarak 3B nesne sınıflandırmasında yüksek doğruluk sağlamaktadır. PointNet++'ın orijinal PointNet'e kıyasla daha iyi sonuçlar sunduğu, özellikle yüksek çözünürlüklü Lidar verilerinde daha etkili olduğu görülmüştür (Qi vd., 2017). He ve arkadaşları (2017) ise görüntü işleme alanında yaygın kullanılan Mask R-CNN algoritmasını Lidar verisi üzerinde uygulayarak 3B nesne tespiti ve segmentasyonu gerçekleştirmiş ve yüksek doğruluk oranlarına ulaşmıştır (He vd., 2017).

Bu çalışmaların devamında Chen ve arkadaşları (2022), derin öğrenmeyi Geniş Öğrenme Sistemi (BLS) ile birleştirerek Point-BLS adlı yeni bir ağ modeli önermiştir. Deneyler, Point-BLS'nin hem doğruluk hem de eğitim süresi açısından klasik derin öğrenme yöntemlerine göre avantaj sağladığını ve ModelNet40 veri setinde %87,48 doğruluk oranına ulaştığını göstermiştir (Chen vd., 2022). Grafik tabanlı yaklaşımlar da literatürde dikkat çekmektedir. Zhang ve Rabbat (2018), PointGCN modeliyle nokta bulutlarını sınıflandırmak için Graph-CNN yapısını önermiştir. Yerelleştirilmiş grafik konvolüsyonlarını farklı havuzlama yöntemleriyle birleştiren bu yapı, ModelNet10 ve ModelNet40 üzerinde rekabetçi sonuçlar elde etmiştir (Zhang & Rabbat, 2018).

Son yıllarda literatüre giren bir diğer yöntem Rao ve arkadaşlarının (2022) geliştirdiği Siyam PointNet++'tır. Bu yöntem nokta çiftleri arasındaki ilişkileri keşfederek sınıflandırma doğruluğunu artırmayı hedeflemektedir. IEEE GRSS Veri Füzyon Yarışması 2019 veri kümesi üzerinde yapılan deneylerde, orijinal PointNet++'a kıyasla daha yüksek doğruluk ve ortalama kesişim (MIOU) değerlerine ulaşılmıştır. Bununla birlikte, azınlık sınıflarındaki dengesizlik sorunları dikkat çekmiştir (Rao vd., 2022). Azizmalayeri ve arkadaşları (2018) ise çekirdek korelasyon tabanlı bir CNN yöntemi önererek MNIST veri

setinde %99,4 gibi yüksek bir doğruluk oranı elde etmiş ve mevcut yöntemlerle benzer seviyede başarı göstermiştir (Azizmalayeri vd., 2018).

Ayrıca, Gong ve arkadaşları (2021) nokta bulutlarının daha iyi temsil edilebilmesi için küresel ve yerel özelliklerin birlikte kullanıldığı bir ağ mimarisi geliştirmiştir. Çalışma sonuçları, küresel özelliklerin entegrasyonunun farklı veri kümelerinde sınıflandırma ve segmentasyon başarısını artırdığını göstermiştir (Gong vd.,2021). Riou ve arkadaşları (2022) ise daha gerçekçi bir senaryoya odaklanarak seyrek ve gürültülü Lidar verilerinde takviyeli öğrenme yönteminin kullanımını önermiştir. Bu yaklaşım, farklı örnekleme yoğunluklarında dahi diğer yöntemlere göre daha tutarlı bir sınıflandırma başarısı sağlamıştır (Riou vd., 2022).

1.3. Lidar Yapısal Bileşenleri

Lidar sensörleri, nesnelerin konumunu belirlemek için lazer darbelerinden yararlanan bir teknolojidir. Günümüzde bu yöntem haritalama, otonom araçlar, robotik sistemler ve uzaktan algılama gibi birçok alanda tercih edilmektedir çünkü yüksek doğrulukta çevresel algı sunabilmektedir.

Bir Lidar sistemi genellikle lazer verici, alıcı, döner mekanizma ve zaman ölçüm bileşenlerinden oluşur. Çalışma prensibi, gönderilen lazer ışınının bir yüzeye çarpıp geri dönmesi ve bu sürenin ölçülmesi esasına dayanır. Ölçülen süre sayesinde objenin sensöre olan uzaklığı hesaplanır ve taranan alanın üç boyutlu modeli elde edilebilir. Lidar'ın ürettiği temel çıktı nokta bulutlarıdır ve günümüzde gelişmiş sensörler sayesinde hem yatay hem de dikey yüzeylerde oldukça yoğun nokta bulutları elde etmek mümkündür (Shan & Toth, 2018).

Elde edilen nokta bulutları, her bir noktanın x, y, z koordinatlarını içerir. Bunun yanında, bazı durumlarda yoğunluk, zaman damgası veya darbe numarası gibi ek bilgiler de sağlanır. Bu veriler kullanılarak çevrenin geometrik yapısı modellenebilir. Ancak sadece verinin toplanması yeterli değildir bu noktaların sınıflandırılması ve anlamlı hale getirilebilmesi için gelişmiş işleme yöntemlerine ihtiyaç duyulmaktadır. Son yıllarda makine öğrenmesi ve derin öğrenme tekniklerinin gelişmesiyle birlikte Lidar verilerinin işlenmesi çok daha etkin bir hale gelmiştir. Böylece sadece üç boyutlu yapının değil, aynı zamanda çevredeki nesnelerin türlerinin de algılanması mümkün olmuştur (Qi vd., 2017).

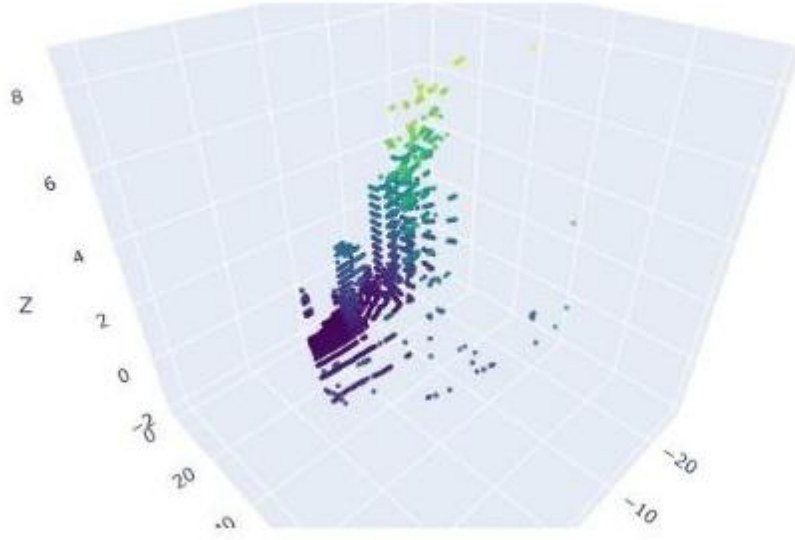
Bu çalışmada kullanılan NuScenes veri seti, Velodyne HDL-32E adlı üç boyutlu Lidar sensörü ile elde edilen nokta bulutlarını içermektedir. Yüksek yoğunluklu tarama kapasitesi ile bilinen bu sensör, özellikle otomotiv sektöründe yaygın olarak tercih edilmektedir. Çalışmada kullanılan veriler, Velodyne HDL-32E sensöründen elde edilen uzamsal bilgileri doğrudan yansıtmakta olup sınıflandırma algoritmalarıyla uyumlu hale getirilerek analiz edilmiştir (Velodyne, 2020).

1.4. Nokta Bulutu Verileri

Nokta bulutu verileri, Lidar sensörlerinden elde edilen ve çevredeki nesnelerin üç boyutlu temsilini sağlayan işlenmemiş veri kümeleridir. Her bir nokta, konum bilgisini ifade eden x , y , z koordinatlarının yanı sıra yoğunluk gibi ek bilgiler de içerir. Bu nedenle nokta bulutları robotik algılama, sınıflandırma, haritalama ve modelleme gibi pek çok uygulamada temel veri formatı olarak kullanılmaktadır (Pomerleau vd., 2015).

Bu tez kapsamında kullanılan nokta bulutu verileri, NuScenes veri setinin Lidarsegmini sürümünden alınmıştır. Bu veriler, Velodyne HDL-32E sensörü ile toplanmış olup, yüksek yoğunluklu üç boyutlu nokta kümeleri oluşturulmuştur. Ancak nokta bulutlarının yapısı düzensiz olduğu için, taramalarda noktaların yoğunluğu, dağılımı ya da nesne biçimleri değişkenlik gösterebilmektedir. Bu durum, düzenli grid yapısındaki veriler için geliştirilen geleneksel bilgisayarla görme yöntemlerinin doğrudan uygulanmasını zorlaştırmaktadır. Bu sebeple, nokta bulutu verilerinin işlenmesi özel ön hazırlık adımları ve özelleştirilmiş derin öğrenme teknikleri gerektirmektedir (Guo vd., 2020).

Veri işleme ve görselleştirme adımları için bu çalışmada Google Colab kullanılmıştır. Colab tercih edilmesinin en önemli nedeni, özellikle yüksek hesaplama gücü gerektiren projelerde sağladığı GPU desteği ve bulut tabanlı yapısıdır. Bunun yanında, Plotly gibi etkileşimli görselleştirme kütüphanelerinin daha sorunsuz çalışabilmesi de önemli bir avantaj sağlamaktadır. Bu nedenlerle, tüm deneysel süreçler Google Colab üzerinde gerçekleştirilmiştir (Bisong, 2019).



Şekil 1. 7 Sınıfa ait ilk on bin nokta

Çalışmanın bu görselleştirme adımında, nuScenes veri kümesinden elde edilen ham verilerin genel yapısını incelemek için 7 sınıfa ait tüm noktalardan rastgele seçilen ilk 10.000 nokta kullanılarak 3B dağılım grafiği oluşturulmuştur. Gerçek Lidar verileri milyonlarca nokta içerdiğinden, görselleştirme aşamasında işlem verimliliğini artırmak amacıyla rastgele seçilmiş 10.000 nokta kullanılmıştır. Bu örneklemede Plotly görselleştirme aracı ile daha hızlı ve kararlı grafik oluşturulması sağlanmıştır (Zhang vd.,2022). Noktaların üç boyutlu uzaydaki dağılımları Google Colab ortamında Python ve Plotly kütüphanesi kullanılarak çizdirilmiştir. Noktalar, yükseklik bilgilerini temsil eden z koordinatına göre renklendirilmiştir. Renklendirme için kullanılan Viridis renk skalası ile düşük değerler koyu mavi ve mor tonlarında, yüksek değerler ise açık yeşil ve sarı tonlarında gösterilerek derinlik algısını artırılmıştır (Hunter, 2007). Bu renklendirme sayesinde zemine yakın nesneler daha koyu renklerde görünürken, yüksek yapılar ya da yüksek nesneler açık renklerle temsil edilmiştir. Bu renkli gösterim ile nokta bulutunun yapısı, yoğunluğu ve dağılımı gibi özellikleri görsel olarak daha anlaşılır bir hale getirilmiştir.

1.4.1. Sınırlayıcı Kutu Yapısı

Lidar tabanlı nesne tanıma ve sınıflandırma çalışmalarında en çok kullanılan yöntemlerden biri sınırlayıcı kutu yapısıdır. Bu kutular, üç boyutlu ortamda bir nesnenin

konumunu, boyutlarını ve yönünü tanımlamaya yarar. Genellikle merkez koordinatları genişlik, uzunluk, yükseklik ve dönüş bilgileriyle ifade edilir. Bu çalışmada kullanılan nuScenes veri seti de her Lidar taramasında sahnedeki nesnelere ait üç boyutlu sınırlayıcı kutuları içermektedir. Her kutunun araba veya yaya gibi bir sınıf etiketi bulunması sayesinde, sahnedeki nesnelerin hangi türden olduğu net bir şekilde belirlenebilmektedir. Sınırlayıcı kutular yalnızca görsel bir açıklayıcılık sunmakla kalmaz, aynı zamanda hangi Lidar noktalarının hangi nesneye ait olduğunun anlaşılmasında da kritik bir rol oynar (Caesar vd., 2020).

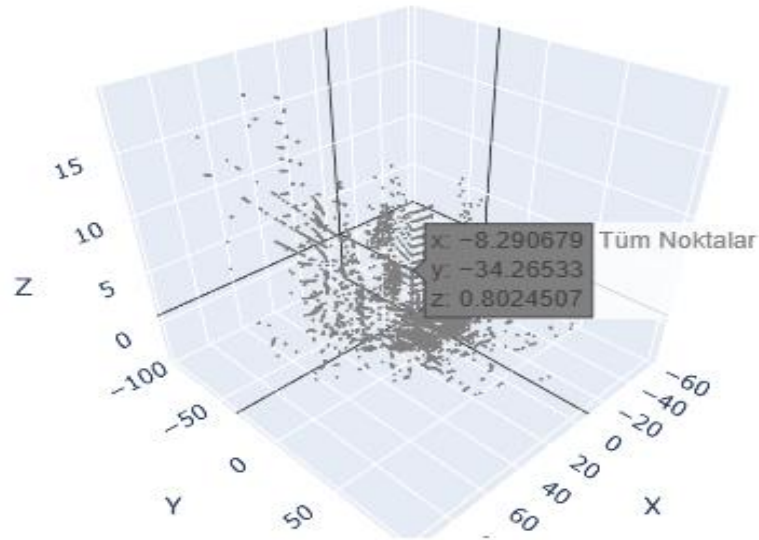
Bu kutular üzerinden yapılan analizler, her sınıfa ait kaç adet kutunun bulunduğunu ve sınıflar arasında dengenin nasıl olduğunu ortaya koymuştur. Bu tür bir kontrol oldukça önemlidir çünkü dengesiz dağılımlar sınıflandırma modellerinin performansını doğrudan etkileyebilir. Örneğin, bazı sınıflara ait veriler diğerlerine göre çok daha fazlaysa, model bu sınıfları öğrenmeye fazla eğilim gösterebilir ve diğer sınıflarda başarısı düşebilir. Bu nedenle veri setinin dengesi, modelin adil ve doğru sonuçlar üretebilmesi açısından kritik bir unsurdur.

Çalışmanın bu adımında nuScenes Lidarseg-mini veri kümesinden elde edilen 3B nokta bulutu verileri kullanılarak, sahne içindeki nesnelerin uzamsal konumları ve sınıf dağılımları görselleştirilmiştir. Bu adımda, örnek bir kutu tanımlanarak bu kutunun içerisine düşen Lidar noktaları tespit edilmiş ve ilgili sınıf etiketlerine göre renklendirilmiştir. Bu işlem farklı sınıfların kutu içerisinde nasıl dağıldığını analiz etmek amacıyla gerçekleştirilmiştir.

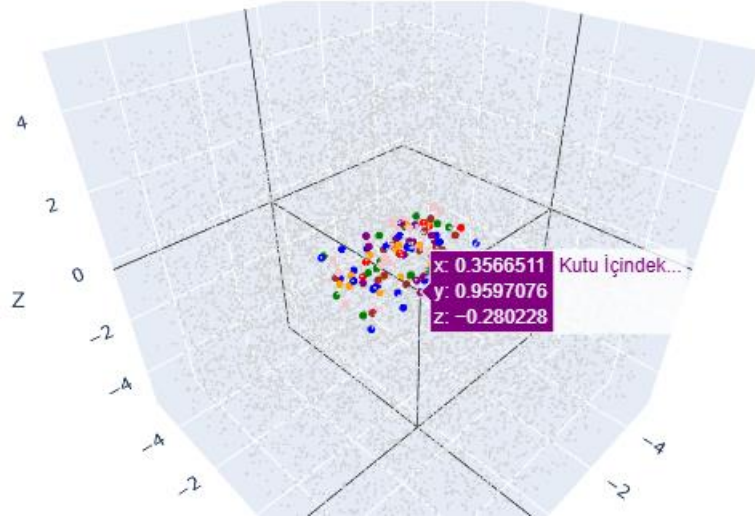
Çalışmanın bu adımında görselleştirme için kullanılan nokta bulutu, rastgele oluşturulmuş 10.000 adet 3B noktadan oluşmaktadır. Bu noktalara ait sınıf etiketleri 0 ile 6 arasında rastgele dağıtılmış olup, her sayı çalışmada kullanılan sınıflardan birini temsil etmektedir. Tablo 1’de sınıf etiketlerinin karşılık geldiği renk eşlemesi sunulmaktadır.

Tablo 1. Sınıf etiketlerinin renk eşleřtirmesi

Sınıf Etiketi	Sınıf Adı	Renk
0	Araba	Mavi
1	Yaya	Kırmızı
2	Otobüs	Turuncu
3	Kamyon	Yeşil
4	Motosiklet	Mor
5	Bariyer	Kahverengi
6	Bisiklet Rafı	Pembe



Şekil 2. Kutu içindeki ham noktalar



Şekil 3. Kutu içinde renkli etiketlenmiş sınıflar

Şekil 2’de sahnedeki tüm Lidar noktalarının gri tonlarda olduğu görülürken ikinci görselde olan Şekil 3’te bir nesneye karşılık gelen 3B kutu içinde kalan noktalar, ait oldukları sınıflara göre renklendirilmiştir. Buradaki renklendirme işlemi, her bir Lidar noktasının sahip olduğu sınıf etiketine göre gerçekleştirilmiştir. Bu renkli gösterim sayesinde, kutu içerisindeki nesneye ait nokta yoğunluğu ve yapısal dağılım daha açık şekilde analiz edilebilmiştir.

Çalışmada veriler, Python programlama dili kullanılarak Google Colab ortamında işlenmiş ve Plotly kütüphanesi yardımıyla görselleştirilmiştir. 3B kutu yapısı, ilgili sahneye ait etiketlerden alınmış olup, kutunun merkezi, boyutları ve dönüşümleri kullanılarak kutu içine düşen noktalar matematiksel olarak filtrelenmiştir. Bu filtreleme ile hedefe ait Lidar noktaları ayrıştırılarak renklendirilmiş ve sınıf etiketleriyle birlikte analiz edilmiştir. Kutu içerisine giren noktalar, geometrik olarak kübik hacim içerisinde kalanlar üzerinden seçilmiştir. Bunun için kutu merkez koordinatları, boyutları ve yönelimi tanımlanmış, ardından bu kutu içerisine düşen noktalar matematiksel olarak filtrelenmiştir. Filtreleme işleminde noktaların kutu eksenine göre konumları hesaplanarak hacim sınırları içerisinde kalanlar ayrıştırılmıştır.

Bu görselleştirme yöntemi hem verinin kalitesini hem de sınıf dağılımını ön analiz aşamasında değerlendirmek için oldukça işlevseldir (Caesar vd., 2020; Hackel vd., 2017).

1.5. Sınıflandırma Performans Ölçütleri

Performans ölçütlerinin doğru bir şekilde seçilmesi Lidar nokta bulutu verilerinin sınıflandırmasında kullanılan öğrenme modellerinin başarısını değerlendirebilmek için kritik bir öneme sahiptir. Bu çalışmada, 7 sınıflı bir sınıflandırma problemi hedeflendiği için, her bir sınıfa ait performans değerlendirmesinin ayrı ayrı yapılması ve modelin genel başarısını ölçmek için toplu metriklerin kullanılması gerekli olmuştur.

1.5.1. Doğruluk

Doğruluk, modelin tüm tahminlerinden kaç tanesinin doğru tahmin olduğunu gösteren bir metriktir.

$$\text{Doğruluk} = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Denklem (1)'de TP (Pozitif Doğru), TN (Negatif doğru), FP (Yanlış Pozitif) ve FN (Yanlış Negatif) kavramları, modelin tahmin ettiği ve gerçek etiketler arasındaki ilişkiyi temsil etmektedir (Sokolova & Lapalme, 2009).

1.5.2. Kesinlik (Precision)

Kesinlik, modelin pozitif olarak belirlediği örnekler arasında, gerçekte doğru olanların oranını ifade etmektedir.

$$\text{Kesinlik} = \frac{TP}{TP+FP} \quad (2)$$

Bu metrik, yanlış pozitif oranının azaltılması istenen uygulamalarda tercih edilir. (Bu çalışmada, özellikle yaya gibi seyrek noktalı sınıflarda modelin hatasız tespiti için hassasiyet değeri kritik bir rol oynamıştır (Powers, 2011).

1.5.3. Duyarlılık/Gerçek Pozitif Oranı

Duyarlılık, gerçek sınıf etiketine sahip örneklerin model tarafından kaç tanesinin doğru sınıflandırıldığını gösterir.

Bu metrik, sınıf içindeki eksik tanımları ön plana çıkarmakta kullanılır. Örneğin, motosiklet gibi az sayıda görülen sınıflarda duyarlılığın yüksek olması modelin nadir nesneleri de iyi tanıyabildiğini gösterir (Powers, 2011).

1.5.4. F1 Skoru

F1 skoru kesinlik ve duyarlılık metrikleri arasında denge kurar ve Denklem (3)'te tanımlanmıştır.

$$F1 = 2 \frac{Kesinlik.Duyarlilik}{Kesinlik+Duyarlilik} \quad (3)$$

F1 skoru, dengesiz veri kümelemesinde daha doğru bir performans göstergesi sunduğu için bu çalışmada, hedeflenen tüm 7 sınıf için hem bireysel F1 skorları hem de ortalama F1 skoru hesaplanmıştır (Powers, 2011).

1.5.5. Karışıklık Matrisi (Confusion Matrix)

Lidar verileri çok sınıflı ve dengesiz veri kümesine sahip oldukları için karışıklık matrisi, sınıflandırma problemlerinde büyük önem taşımaktadır. Sadece toplam doğruluk değeri yeterli bilgi sağlamayabilir ve bu değer yüksek olsa bile bazı sınıflarda ciddi hatalar olabilir. Bu nedenle, özellikle az örneğe sahip sınıfların model tarafından doğru bir şekilde sınıflandırılıp sınıflandırılmadığını analiz edebilmek için karışıklık matrisi kullanılmaktadır. Örneğin çalışmada kullanılan modelin kamyon sınıfına ait noktaları sıklıkla araba olarak tahmin etmesi, karışıklık matrisiyle gözlemlenebilir ve bu durum, sınıflar arasındaki görsel veya yapısal benzerliklerden dolayı olabilir.

Çalışmada kullanılan modelin hangi sınıflarda güçlü performans sergilediği ve hangi sınıflarda hata oranlarının yüksek olduğu karışıklık matrisi ile sistematik olarak analiz edilmiştir. (Sokolova & Lapalme, 2009; Powers, 2011).

Bu metriklerin her biri sayesinde modelin sadece genel başarısını değil, ayrı ayrı sınıflar üzerindeki performansının da değerlendirilmesine olanak sağlamıştır.

2. YAPILAN ÇALIŞMALAR

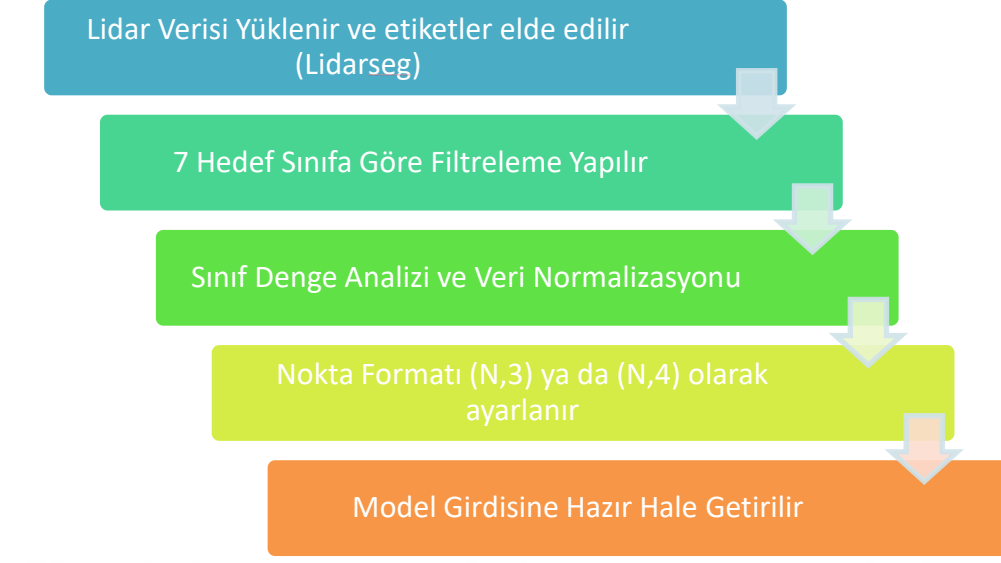
Bu bölümde, Lidar nokta bulutu verilerinin sınıflandırılması için izlenen yöntemsel yaklaşımlar detaylı olarak sunulmaktadır. Bu çalışmada gerçek Lidar verisi kullanılarak farklı derin öğrenme yöntemleri ile klasik makine öğrenmesi algoritmalarının Lidar verisi üzerinde sınıf bazlı performans değerlendirmesi yapılmıştır.

2.1. Veri Kümeleri ve Etiketleme Yapısı

Bu çalışmada kullanılan veri kümesi nuScenes veri kümesinin mini sürümüdür. NuScenes, çoklu Lidar sensörlerinden elde edilen üç boyutlu nokta bulutlarını ve çeşitli yönlerden alınmış kamera görüntülerini bir araya getiren bir veri kümesidir. Çalışmada her bir noktanın ait olduğu nesne sınıfı Lidar verileriyle birlikte gelen Lidarseg etiket dosyaları kullanılarak belirlenmiştir (Caesar vd., 2020).

2.2. Veri Önışleme Adımları

İşlenmemiş Lidar verileri, uzayda üç boyutlu x,y,z koordinatları ve bazı durumlarda yoğunluk bilgisi içeren noktalardan oluşur. Bu veriler yüksek hacimli, düzensiz ve seyrek yapılıdır olduğu için makine öğrenmesi modellerine direkt uygulanmadan önce birkaç önışleme adımlarından geçirilmesi gerekmektedir (Hackel vd., 2017).



Şekil 4. Lidar noktalarının işlenme süreci

Şekil 4'te Lidar noktalarının işlenme süreci şema grafiği ile gösterilmiştir. Çalışmada öncelik olarak veriler yüklenmiş, daha sonra sadece hedef sınıflar filtrelenmiş ve bu sınıflar için dengeli veri sağlanmıştır. Son olarak uygun formatta giriş verisi oluşturulmuştur.

2.2.1. Etiketli Noktaların Elde Edilmesi

Bu çalışmada kullanılan nuScenes veri kümesi her Lidar noktasına ait segmentasyon etiketleri de içermektedir. Lidarseg adlı özel dosyalarda bu etiketler tutulmakta olup her bir Lidar taramasına karşılık gelen .bin dosyaları içinden yüklenerek her noktanın ait olduğu sınıf bilgisi elde edilmiştir (Caesar vd., 2020).

2.2.2. Sınıf Filtreleme ve Hedef Seçimi

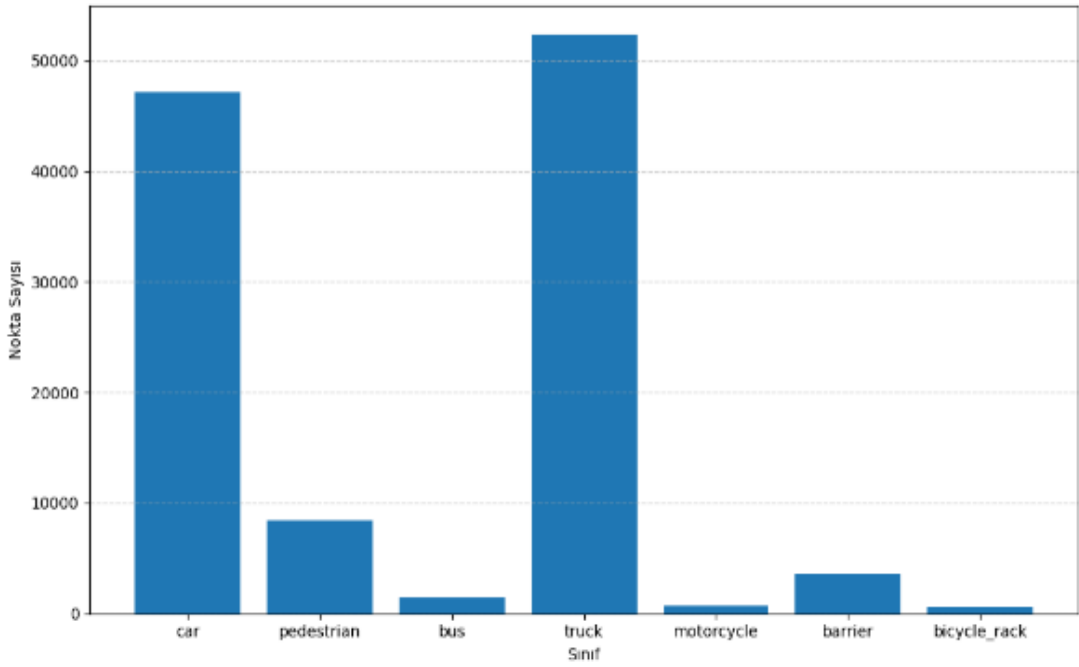
NuScenes verisinde toplam 23 farklı sınıf bulunmaktadır ancak bu çalışmada yalnızca hedeflenen yedi nesne sınıfı analizlenmiştir ve bu sınıfların dışındaki tüm noktalar analiz dışı bırakılmıştır. Bu sınıflar, motorlu kara taşıtları örneğin otomobil, kamyon, otobüs, motosiklet, yayalar ve belirli sabit ya da taşınabilir nesneler olarak belirlenmiştir. Bu sınıflar seçilirken sahne içerisinde yaygın görülmeleri, veri miktarının yeterliliği, uygulama bağlamında anlamlılık kriterleri dikkate alınmıştır.

- Sahne içerisinde yaygın görülmeleri: Çalışmada özellikle otomobil, otobüs ve yaya sınıfları gibi sahnelerde sıklıkla yer alan nesneler tercih edilmiştir. Bu sınıflar veri setinde yoğun şekilde bulunmaları sayesinde modelin genelleme başarısını arttırmaktadır (Alaba & Ball, 2022).
- Veri miktarının yeterliliği: Veri setindeki bazı sınıflar örneğin bisiklet rafı veya motosiklet çok az sayıda örneğe sahip olduğundan dolayı bu tür sınıflar modelin aşırı öğrenmesine veya öğrenememesine neden olabilir. Küçük sınıfların temsil gücünü arttırmak için veri çoğaltma yapılabilmektedir (Griffithd & Boehm, 2019).
- Uygulama bağlamında anlamlılık: Trafik ortamlarında kritik öneme sahip sınıfların seçimi, yapılan sınıflandırmayı daha anlamlı ve uygulamaya dönük hale getirmektedir (Lisboa vd., 2025).

Bu filtreleme sonucunda, az sayıda noktaya sahip olan sınıflar çalışmaya dahil edilmemiştir. Bu sayede hem model karmaşıklığı azaltılmış hem de sınıflar arası ayırt edilebilirlik artırılmıştır.

Tablo 2. Hedeflenen 7 sınıf

Sınıf ID	Sınıf Adı	Üst Kategori
14	Araba	Araç
17	Kamyon	Araç
12	Otobüs	Araç
16	Motosiklet	Araç
1	Yaya	İnsan
6	Bariyer	Hareketli Nesne
10	Bisiklet rafı	Sabit Nesne



Şekil 5. Sınıf başına nokta sayılarının histogramı

Şekil 5’te sunulan histogram grafiğinde önışleme adımlarından sonra her sınıfa ait kalan Lidar noktalarının dağılımı görsel olarak sunulmaktadır. Veri kümesinde yapılan filtreleme işlemi sonucunda korunan yedi hedef sınıfa ait örnek sayıları aşağıdaki gibidir:

- Araba = 47,184 nokta
- Kamyon = 52,366 nokta
- Otobüs = 1,435 nokta
- Motosiklet = 711 nokta
- Yaya = 8,448 nokta
- Bariyer = 3,631 nokta
- Bisiklet rafı = 591 nokta

Sınıflara ait nokta sayıları incelendiğinde bazı sınıflar arasındaki nokta sayılarında gözle görülür bir fark olduğu görülmektedir. Örneğin kamyon ve araba sınıfları toplam nokta sayısının büyük bir kısmını oluştururken, bisiklet rafı veya motosiklet gibi sınıfların içerdiği nokta sayısı daha düşüktür. Bu durum sınıf dengesizliği problemini oluşturabileceğinden, çalışmanın eğitim sürecinde alt örnekleme ve üst örnekleme tekniklerinin birlikte kullanılmıştır.

2.2.3. Sınıf Denge Analizi ve Veri Normalizasyonu

Bu çalışmada kullanılan Lidar nokta bulutu verileri, sınıflandırma modellerine girdi olarak kullanılmadan önce kapsamlı bir ön işleme sürecinden geçirilmiştir. İlk adımda, veriler NumPy kütüphanesi yardımıyla uygun sayısal formata dönüştürülmüş, ham veriler `np.array()` fonksiyonu aracılığıyla modele uygun bir yapı haline getirilmiştir. Noktaların uzamsal dağılımlarındaki ölçek farklarını azaltmak amacıyla, her bir x, y, z koordinat bileşeni normalize edilmiştir. Normalizasyon işlemi sırasında veri kümesinin ortalaması ve standart sapması `np.mean()` ve `np.std()` fonksiyonları kullanılarak hesaplanmıştır ve tüm değerler bu istatistiklere göre yeniden ölçeklendirilmiştir. Bu sayede, modelin eğitim sürecinde belirli bir eksene aşırı duyarlılık göstermesi engellenmiş ve öğrenme süreci stabilize edilmiştir.

Veri kümesindeki sınıf dağılımı incelendiğinde, örneğin araba ve kamyon gibi sınıflarda gözle görülür şekilde daha fazla örnek bulunduğu, buna karşın bariyer ve bisiklet rafı gibi sınıfların örnek sayısının oldukça düşük olduğu tespit edilmiştir. Bu dengesizliğin model performansını olumsuz etkilememesi için stratified sampling yöntemi kullanılmıştır. Bu yöntemle, eğitim, doğrulama ve test kümeleri oluşturulurken her sınıfın oransal temsili korunmuştur. Böylece modelin özellikle az örnek içeren sınıflarda da anlamlı bir öğrenme gerçekleştirmesi hedeflenmiştir.

Veri ayrımı işlemi sonrasında her kümede etiketlerin dağılımı `np.unique()` fonksiyonu ile kontrol edilmiştir. Elde edilen istatistikler, sınıf dengesi açısından tüm alt kümelerin ana veri setine benzer oranlarda temsil edildiğini göstermiştir. Ayrıca eğitim ve test kümeleri birleştirilirken `np.concatenate()` fonksiyonu kullanılmış, tüm verilerin tek bir dizi yapısında tutarlı biçimde işlenmesi sağlanmıştır. Bu adımlar hem modelin genelleme kabiliyetini artırmak hem de karşılaştırmalı algoritma analizlerinde adil bir değerlendirme ortamı oluşturmak amacıyla uygulanmıştır. Böylece, klasik makine öğrenmesi yöntemleri ile derin öğrenme tabanlı yaklaşımların aynı koşullar altında test edilmesi mümkün olmuştur.

2.2.4. Veri Formatı ve Dönüşümü

Veri formatı makine öğrenmesi ve derin öğrenme tabanlı sınıflandırma algoritmalarında modelin başarısı üzerinde doğrudan etkili bir faktördür. Bu çalışmada kullanılan Lidar verisi, işlenmemiş haliyle üç boyutlu nokta bulutları biçimindedir ve her

bir nokta genellikle x , y , z konum bilgisini taşımaktadır. Ancak bazı durumlarda veriye yoğunluk değeri de eşlik etmektedir.

Bu kapsamda, çalışmada iki farklı veri temsil formatı oluşturulmuştur:

(N, 3) formatı: Bu yapı yalnızca nokta bulutlarının x , y ve z koordinatlarını içermektedir. Nokta sayısı NNN olan bir bulut için her nokta sadece üç sayısal özellik ile tanımlanır. Bu format, sade yapılı model örneği olan temel MLP mimarilerinde tercih edilmektedir (Qi vd., 2017).

(N, 4) formatı: Veriye yoğunluk değerinin de dahil edilmesiyle her nokta dört boyutlu bir hale gelmektedir. Bu format, özellikle fiziksel ortamdan elde edilen sinyal gücünün ayırt edici özellik olarak kullanılabilir ve CNN temelli yapılarda faydalı olmaktadır (Zhou vd., 2018).

Verilerin bu şekilde dönüştürülmesi ile farklı algoritmalar arasında adaletli bir karşılaştırma yapabilmektedir. Yoğunluk bazı sınıfların arasında ayırım yapılmasını kolaylaştırabilirken, bazı durumlarda modele aşırı yük bindirerek öğrenme sürecini olumsuz etkileyebileceği için çalışmada hem üç boyutlu hem de dört boyutlu veri temsilleri hazırlanmış ve böylece model mimarisine uygun biçimde esnek kullanım sağlanmıştır. Veriler, numpy dizileri olarak saklanmış ve standartlaştırılmış girdi formatları hâline getirilmiştir. Bu işlem, veri yükleme, temizleme ve filtreleme aşamalarının ardından uygulanmış, gerekli format dönüşümleri gerçekleştirilerek model girişine doğrudan uygun hale getirilmiştir (Geiger vd., 2013; Wu vd., 2019).

Tablo 3. Veri Formatları

Özellik	(N, 3) Formatı	(N, 4) Formatı
Boyut	N x 3	N x 4
İçerik	x, y, z	x, y, z, yoğunluk
Kullanım Alanı	Temel koordinat tabanlı modeller	CNN gibi ek özellik isteyen modeller
Bellek Gereksinimi	Daha düşük	Bir miktar daha yüksek
Hesaplama Maliyeti	Göreceli olarak daha az	Özellik sayısı arttığı için biraz daha fazla
Model Duyarlılığına Etkisi	Düşük/Orta	Genellikle detaylı sahnelerde pozitif etki
Uygun Olduğu Model Türleri	MLP gibi sınıflayıcılar	CNN, bazı derin ağlar, yoğunluk duyarlı yapılar

Nokta bulutu biçiminde gelen veriler Lidar verilerinin sınıflandırma modellerine uygun hale getirilebilmesi için, yeniden yapılandırılarak iki temel veri formatında kullanılmıştır. Sade (N, 3) biçimi yalnızca uzamsal koordinat bilgilerini içerirken genişletilmiş (N, 4) formatı ise nokta başına yoğunluk bilgisini de barındırır. Bu dönüşümün amacı, farklı model türlerinin veri ihtiyaçlarına uyum sağlamak ve zengin özelliklerin sınıflandırma başarısına etkisini incelemek olmuştur (Fan vd., 2022). Örneğin bazı derin öğrenme mimarileri yalnızca konumsal verilerle etkili çalışabilirken, bazı algoritmalar yoğunluk değerlerinden de fayda sağlayabilir. Her iki formatın temel farkları ve kullanım amaçları Tablo 3'te özetlenmiştir.

Bu karşılaştırmalı yapı sayesinde, kullanılan modellerin hangi veri tipinde daha başarılı sonuçlar verdiği gözlemlenebilmiş, ayrıca yoğunluk bilgisinin sınıflandırma performansına katkısı da deneysel olarak değerlendirilmiştir.

2.3. Makine Öğrenmesi Yaklaşımları

Bilgisayarların programlanmadan direkt veriden öğrenmesini sağlayan yapay zekâ alanlarından biri de makine öğrenmesidir. Makine öğrenmesinde temel amaç, geçmiş verilerin çıktılarını kullanarak yeni gelen veriler için doğru tahminler yapabilmektir. Klasik makine öğrenmesi yöntemleri genellikle istatistiksel analiz, karar ağaçları, mesafe ölçümleri veya olasılık temelli modeller üzerine kuruludur. (Bishop, 2006).

Bu çalışmada makine öğrenmesi yöntemlerinin tercih edilme sebeplerinden biri, Lidar nokta bulutlarının belirli özellikler üzerinden hızlı bir şekilde sınıflandırılabilmesi olmuştur. Derin öğrenme yöntemlerine kıyasla daha az işlem gücü gerektirmeleri, model kurulumunun daha basit olması ve eğitim sürelerinin genellikle daha kısa olması avantajları arasında yer almaktadır (Geron, 2019).

Bu çalışmada makine öğrenmesi kapsamında Random Forest, ML-MLP ve K-En Yakın Komşu (KNN) algoritmaları uygulanmıştır. Random Forest, birden fazla karar ağacının sonuçlarını birleştirerek daha kararlı ve doğru tahminler üreten öğrenme yöntemidir (Breiman, 2001). KNN yöntemi ise veri noktaları arasındaki mesafeyi hesaplayarak en yakın komşuların sınıflarına göre tahmin yapmaktadır (Zhang, 2016). MLP ise, katmanlı yapısı sayesinde doğrusal olmayan ilişkileri modelleyebilen yapay sinir ağı tabanlı bir yaklaşım olup, bu çalışmada hem makine öğrenmesi kategorisi altında hem de derin öğrenmede değerlendirilmiştir (Raschka ve Mirjalili, 2020).

Bu algoritmaların uygulanması sırasında, Lidar verisinin yüksek hacmi dikkate alınmış ve tüm 7 sınıf için tam veri seti üzerinde çalışılmıştır.

2.3.1. Random Forest (Rastgele Orman) Algoritması

Bu çalışmada kullanılan yöntemlerden biri, bir makine öğrenmesi algoritması olan Random Forest yöntemidir. Random Forest sınıflandırma algoritması çok sayıda karar ağacının bir araya gelmesiyle oluşan topluluk temelli algoritmadır. Her karar ağacı, veri üzerinde bağımsız şekilde öğrenim gerçekleştirirken, son sınıf tahmini tüm ağaçların oy birliği ile belirlenmektedir (Breiman, 2001). Bu algoritma özellikle yüksek boyutlu ve düzensiz veri kümelerinde genellikle başarılı sonuçlar vermesiyle bilinmektedir. Nokta bulutu gibi sınıf sınırlarının net olmadığı ve gürültülü veri tiplerinde, RF algoritmasının esnekliği avantaj sağlamıştır. Modelin girdisi, her bir noktanın x, y, z koordinatları ile

yoğunluk bilgisini içeren (N, 4) boyutunda bir veri kümesidir. Burada N, toplam etiketlenmiş nokta sayısını ifade etmektedir.

Random Forest modeli sklearn kütüphanesi kullanılarak uygulanmıştır. Model 100 adet karar ağacından oluşmakta ve her bir ağacın derinliği en fazla 25 olacak şekilde sınırlandırılmıştır. Bu parametre seçimi, aşırı öğrenmeyi engellemek ve daha hızlı eğitim sağlamak amacıyla tercih edilmiştir.

Tüm veri seti, sklearn train_test_split fonksiyonu kullanılarak %80 eğitim ve %20 test olmak üzere ayrılmıştır. Stratified sampling yöntemi kullanılarak sınıflar arası denge korunmuştur.

Model, eğitim verisi üzerinde fit() fonksiyonu ile eğitilmiş ve tahminler predict() fonksiyonu ile gerçekleştirilmiştir. Değerlendirme sürecinde doğruluk skoru ile genel doğruluk, sınıflandırma raporu ile sınıf bazlı kesinlik, duyarlılık ve F1 skorları hesaplanmıştır.

Model, test veri kümesi üzerinde yaklaşık %97 doğruluk elde etmiş ve 7 sınıf için yüksek seviyede dengeli başarı göstermiştir. Bu sonuçlar hem genel doğruluğu hem de sınıflar arası dağılımı dikkate alan sınıflandırma raporu çıktısıyla detaylandırılmıştır.

Random Forest algoritması, geometrik özellikleri temel alan bir sınıflandırma probleminde oldukça başarılı performans sergilemiştir. Özellikle sınıf sayısının çokluğu ve veri kümesinin düzensizliği ve göz önüne alındığında, modelin hızlı eğitilmesi ve yüksek doğrulukla sınıfları ayırt edebilmesi önemli bir avantaj olarak öne çıkmaktadır (Louppe, 2014).

Ancak karar ağaçları yüksek boyutlu verilerde belleği daha fazla kullandığı ve yeterince hızlı tahminler üretmediği için çalışmanın ilerleyen kısımlarında daha yüksek soyutlama yeteneğine sahip olan yapay sinir ağı temelli yöntemlerle karşılaştırmalı analiz yapılmıştır.

2.3.2. Multi-Layer Perceptron (Çok Katmanlı Algılayıcı) (ML-MLP)

Bu çalışmada nokta bulutu verilerinin sınıflandırılması için kullanılan yöntemlerden biri de MLP olmuştur. MLP, giriş verilerini tam bağlantılı birçok katmandan geçirerek doğrusal olmayan ilişkileri modelleme yeteneğine sahip bir yöntemdir. MLP'nin tercih edilmesindeki temel neden, yapısının sade ve uygulanabilir olmasıdır. Nokta bulutundan

elde edilen koordinat bilgileri x , y , z değerleri modele doğrudan giriş olarak verilerek sadece temel geometrik özelliklerin sınıflandırmadaki katkısı ölçülmüştür.

Veri hazırlığında, tüm noktalar %80 eğitim – %20 test oranında stratified sampling yöntemi kullanılarak ayrılmıştır. Bu yöntem sayesinde her sınıfın oransal dağılımı hem eğitim hem de test alt kümelerinde korunarak dengesiz sınıf dağılımlarından kaynaklanabilecek yanlışlıkların önüne geçilmiştir. Deneylerin tekrarlanabilirliği için rassal sayı üretici sabitleştirilmiştir.

Nokta bulutu verisinde motosiklet, bisiklet rafı gibi bazı sınıfların diğerlerine göre az sayıda örneğe sahip olması sınıf dengesizliği sorununu ortaya çıkarmıştır. Sklearn kütüphanesindeki MLP uygulaması doğrudan sınıf ağırlıklandırmayı desteklemediği için eğitim setinde aşırı örnekleme uygulanmıştır. Bu yöntemle örneği az sınıflar çoğaltılarak eğitim sırasında her sınıfın eşit oranda temsil edilmesi sağlanmıştır. Test verisinde ise orijinal hali korunarak bağımsız değerlendirme sağlanmıştır. Aşırı örnekleme az örnekli sınıfların duyarlılığını artırsa da yanlış pozitiflerin çoğalmasına yol açabileceğinden kesinlik üzerinde olumsuz etkilere neden olabilmektedir (Chawla vd., 2002).

MLP yöntemi iki gizli katmandan oluşacak şekilde tasarlanmıştır. İlk katmanda 128, ikinci katmanda 64 nöron yer almakta olup her iki katmanda da ReLU (Rectified Linear Unit) aktivasyon fonksiyonu kullanılmıştır. Çıkış katmanında ise yedi nöron bulunmakta ve bunlar sırasıyla araba, yaya, otobüs, kamyon, motosiklet, bariyer ve bisiklet rafı sınıflarını temsil etmektedir. Çok sınıflı yapının gereği olarak, çıkışta softmax aktivasyon fonksiyonu kullanılmıştır.

Eğitim sürecinde optimizasyon algoritması olarak Adam tercih edilmiştir. Kayıp fonksiyonu olarak çok sınıflı sınıflandırma problemlerinde standart yöntem olan Cross-Entropy Loss kullanılmıştır. Ayrıca aşırı uyum riskini azaltmak amacıyla erken durdurma stratejisi uygulanmıştır. Bu kapsamda, eğitim setinin %10'u doğrulama için ayrılmış ve doğrulama kaybında iyileşme görülmeyen 10 ardışık iterasyon sonrasında eğitim durdurularak en iyi ağırlıklar korunmuştur.

Deneyisel tasarımıda, MLP modeli farklı iterasyon sayıları (100, 200) ile eğitilerek eğitim süresinin model performansına etkisi incelenmiştir. Bu sayede daha uzun eğitimlerin doğruluk, kesinlik, duyarlılık ve F1 skorları üzerindeki etkileri gözlemlenmiştir ve az örnekli sınıflardaki performans farklılıkları analiz edilmiştir.

Çalışmanın bu kısmında kullanılan makine öğrenmesi tabanlı MLP modeli yalnızca nokta bulutunun temel koordinat bilgilerini kullanarak hızlı ve düşük maliyetli bir

sınıflandırma yapmamıza olanak sağlamıştır. Ancak, elde edilen sonuçlar incelendiğinde modelin bazı sınıflarda yüksek başarıya ulaşılırken, özellikle örnek sayısı düşük olan sınıflarda performans kayıpları yaşandığı görülmüştür. Bu durum MLP mimarisinin nokta bulutundaki karmaşık ve çok boyutlu ilişkileri öğrenmede yetersiz kaldığını ortaya koymaktadır.

Bu noktada çalışmanın kapsamını genişletmek amacıyla derin öğrenme tabanlı MLP (DL-MLP) modeline geçilmiştir. DL-MLP, klasik MLP'nin yapısının daha derin ve gelişmiş bir versiyonu olarak düşünülebilir. Çok katmanlı tam bağlantılı yapısına ek olarak dropout ve layer normalization gibi düzenleme tekniklerinin kullanılması ile modelin aşırı uyum riskini azaltılmış ve genelleme kapasitesi artırılmıştır.

Böylelikle, çalışmada MLP iki farklı düzeyde incelenmiştir ilki klasik makine öğrenmesi bağlamında, ikincisi ise derin öğrenme yöntemleri arasında değerlendirilmiştir. DL-MLP ile daha güçlü bir modelleme yaklaşımı geliştirilerek sınıflar arasındaki dengesizlik ve karmaşık geometrik ilişkilerin daha iyi yakalanması hedeflenmiştir.

2.3.3. K-En Yakın Komşu (K-Nearest Neighbors, KNN) Yöntemi

Bu çalışmada kullanılan makine öğrenmesi yöntemlerinden biri de K-En Yakın Komşu (KNN) algoritmasıdır. KNN sınıflandırma işlemini gerçekleştirirken veriye dayalı bir yaklaşım benimseyen ve eğitim aşamasında açık bir model oluşturmayan, bunun yerine tüm eğitim verisini bellekte saklayarak yeni bir örneğin sınıfını belirlerken en yakın komşularına bakarak karar veren bir algoritmadır. Bu yöntem, özellikle veri noktalarının uzayda yakınlık ilişkilerinin anlamlı olduğu durumlarda etkili sonuçlar vermektedir (Murphy, 2022).

Girdi Verisi: Modelin girdisi, her bir Lidar noktasının x, y, z koordinatlarını içeren (N, 3) boyutlu veri kümesidir. Burada N, tüm etiketlenmiş nokta sayısını ifade etmektedir.

Model Parametreleri: sklearn kütüphanesinin KNeighborsClassifier sınıfı kullanılmıştır. Komşu sayısı $k=3$ olarak belirlenmiştir. Bu değer, farklı denemeler sonucunda doğruluk ve işlem süresi arasında en iyi dengeyi sağlayan parametre olmuştur. Mesafe ölçümü olarak Öklid mesafesi kullanılmıştır.

Veri Ayırımı: Tüm veri seti, sklearn train_test_split fonksiyonu ile %80 eğitim ve %20 test olarak ayrılmış, stratified sampling yöntemi ile sınıf dengesinin korunmasına dikkat edilmiştir.

Eğitim ve Tahmin: KNN algoritması doğrudan tüm eğitim verisini sakladığından fit aşaması hızlı gerçekleşmiştir ancak test aşamasında her yeni veri noktası için mesafe hesaplanması gerektiği için tahmin süresi daha uzun olmuştur. Tahminler predict() fonksiyonu ile gerçekleştirilmiştir.

KNN modeli, 7 sınıflı tüm nokta bulutu verisi üzerinde yaklaşık %96 doğruluk oranı elde etmiştir. Sınıf bazında, araç ve yaya gibi büyük veri yoğunluğuna sahip sınıflarda yüksek başarı sağlanırken, örnek sayısı az olan motosiklet sınıfında diğerlerine nazaran daha düşük başarı elde edilmiştir. Bu durum, KNN'nin nadir sınıflarda komşuluk bazlı öğrenme kısıtları ile ilişkilidir.

KNN algoritması veri dağılımı hakkında herhangi bir varsayıma ihtiyaç duymadan çalışmıştır. Ancak özellikle nokta sayısının yüksek olduğu bu çalışmada eğitim verisinin tamamının bellekte tutulması ve her tahminde mesafe hesaplamalarının yapılması zaman kaybına yol açmıştır (Peterson, 2009).

2.4. Derin Öğrenme Yaklaşımları

Derin öğrenme çok katmanlı yapay sinir ağları kullanarak verinin karmaşık yapısını çok katmanlı yapay sinir ağları kullanarak otomatik olarak öğrenebilen bir yaklaşımdır. Özellik çıkarımı sürecini insan müdahalesi olmadan kendisinin yapabilmesi klasik makine öğrenmesi yöntemlerinden en temel farkıdır. Özellikle büyük veri setleri, yüksek boyutlu verilerin kullanıldığı durumlarda derin öğrenme yöntemleri sıklıkla kullanılır (Goodfellow vd., 2016).

Bu çalışmada derin öğrenme kapsamında MLP, CNN ve PointNet-Mini kullanılmıştır. MLP, bağlantılı katmanlardan oluşan bir yapay sinir ağı türüdür. Bu model, giriş verilerini katmanlar aracılığıyla işler ve çıktı katmanında sınıflandırma yapar. Özellikle sayısal verilerin doğrudan işlenmesinde etkili olup, bu çalışmada Lidar noktalarının koordinat bilgileri üzerinden sınıflandırma amacıyla kullanılmıştır (GoodFellow vd., 2016).

CNN ise genellikle görsel verilerde tercih edilen bir yöntem olmasına rağmen, 3B nokta bulutlarında da uzamsal ilişkilerin öğrenilmesinde başarılıdır. Model, evrişim katmanları sayesinde verideki lokal desenleri yakalayabilir. Ancak bu çalışmada CNN'in uygulanması sırasında, tüm veri seti üzerinde eğitim yapıldığında yüksek RAM kullanımı

ve uzun eğitim süreleri gibi zorluklarla karşılaşmıştır. Bu nedenle bazı denemelerde nokta sayısı 10.000 ile sınırlandırılmıştır (Qi vd., 2017).

Derin öğrenme yöntemlerinin en büyük avantajı, karmaşık verilerde yüksek doğruluk oranları sunabilmesidir. Ancak deavantajı daha fazla işlem gücü, daha uzun eğitim süresi ve donanım gereksinimi ortaya çıkar. Bu çalışmada MLP, özellikle veri boyutunun tamamı ile çalışırken yüksek doğruluk sağlamıştır CNN ise sınırlı veri üzerinde daha düşük doğruluk verse de mekânsal ilişkileri modelleme potansiyeli nedeniyle yöntemler arasında yer almıştır (LeCun vd., 2015).

2.4.1. Derin Öğrenme MLP (DL-MLP) Yöntemi

Bu çalışmada nokta tabanlı sınıflandırma problemini çözmek amacıyla, makine öğrenmesi kapsamında uygulanan MLP yaklaşımına ek olarak derin öğrenme tabanlı Multi-Layer Perceptron (DL-MLP) modeli de tasarlanmıştır. DL-MLP, tam bağlantılı katmanlardan oluşan bir sinir ağıdır ancak klasik MLP'ye kıyasla daha derin katman yapısı, GPU hızlandırmalı eğitim imkânı ve büyük veri kümelerini daha verimli işleyebilmesi nedeniyle tercih edilmiştir (Goodfellow et al., 2016).

Modelin seçilmesindeki temel gerekçelerden biri, önceki MLP uygulamasında görülen CPU tabanlı eğitim süresinin çok uzun olmasıdır. Scikit-learn tabanlı modelde GPU desteğinin olmaması, eğitim süresini saatler mertebesine çıkarmıştır. Bu nedenle PyTorch tabanlı DL-MLP modeli tercih edilmiştir. Böylelikle model hem GPU hızlandırmalı eğitime olanak tanımış hem de derin mimari tasarımı sayesinde daha yüksek doğruluk değerlerine ulaşmıştır.

DL-MLP mimarisi, giriş katmanında her bir noktanın x, y, z koordinatlarını alacak şekilde tasarlanmıştır. İlk gizli katmanda 256, ikinci gizli katmanda 128, üçüncü gizli katmanda ise 64 nöron bulunmaktadır. Tüm gizli katmanlarda doğrusal olmayan ilişkileri modelleyebilmek için ReLU aktivasyon fonksiyonu kullanılmıştır. Çıkış katmanında yedi nöron yer almakta ve bunlar sırasıyla araba, yaya, otobüs, kamyon, motosiklet, bariyer ve bisiklet rafı sınıflarını temsil etmektedir. Modelin çıktısı, her bir noktanın ilgili sınıfa ait olma olasılığını gösteren bir olasılık vektörüdür.

Eğitim sürecinde Adam optimizasyon algoritması kullanılmıştır (Kingma & Ba, 2014). Kayıp fonksiyonu olarak Cross-Entropy Loss tercih edilmiş, böylece çok sınıflı sınıflandırma problemleri için uygun bir yapı elde edilmiştir. Veri seti, %80 eğitim ve %20

test olacak şekilde bölünmüş ve sınıf dengesinin korunabilmesi için stratified sampling uygulanmıştır.

Modelin uygulaması Google Colab ortamında, GPU hızlandırması kullanılarak gerçekleştirilmiştir. CPU ile yapılan klasik MLP eğitime kıyasla, GPU desteği sayesinde eğitim süresi ciddi ölçüde kısalmıştır. Örneğin, aynı veri setinde klasik MLP yaklaşık 70-80 dakika eğitim süresi gerektirirken, DL-MLP modeli GPU ile ortalama 8–10 dakikada eğitilebilmiştir. Bu durum, Colab ortamında derin öğrenme yöntemlerinin uygulanabilirliğini göstermektedir.

2.4.2. Convolutional Neural Network (CNN)

Bu çalışmada Lidar nokta bulutlarının sınıflandırılmasında kullanılan yöntemlerden biri de Evrişimli Sinir Ağları olmuştur. CNN'ler, özellikle görüntü işleme alanında yaygın kullanılan ve güçlü özellik çıkarma kabiliyetleriyle öne çıkan derin öğrenme modelleridir. Ancak nokta bulutları düzenli piksel matrisleri veya hacimsel gridlerden farklı olarak düzensiz ve seyrek bir yapıya sahiptir. Bu nedenle klasik 2B ya da 3B CNN'lerin doğrudan uygulanması çoğu zaman mümkün değildir.

Bu çalışmada daha yalın ve verimli bir çözüm tercih edilmiştir. Nokta bulutundaki her bir noktanın yalnızca üç boyutlu x, y, z koordinat bilgileri modele giriş olarak verilmiş ve doğrudan 1B CNN yapısı kullanılmıştır. Bu yaklaşım, karmaşık ön işleme adımlarını ortadan kaldırırken modelin noktalar arasındaki temel geometrik ilişkileri öğrenmesine olanak tanımıştır (Guo vd., 2020).

Kullanılan veri kümesi, nuScenes Lidarseg-mini alt setinden alınmıştır. Her bir nokta üç koordinat bilgisiyle temsil edilmiştir. Veriler eğitim (%64), doğrulama (%16) ve test (%20) olacak şekilde stratified sampling yöntemiyle ayrılmış, böylece sınıfların oransal dağılımı tüm alt kümelerde korunmuştur. Ayrıca sınıflar arasındaki ciddi dengesizlik nedeniyle modelin öğrenme sürecini dengelemek amacıyla sınıf ağırlıkları hesaplanarak eğitim sırasında kullanılmıştır.

CNN'in giriş boyutları, her noktanın üç koordinatını işleyebilmesi için (N, 3, 1) olacak şekilde düzenlenmiştir. Burada N, toplam nokta sayısını, 3, koordinat özelliklerini ve 1 ise kanal sayısını ifade etmektedir. Bu yapı sayesinde her nokta, CNN filtreleri tarafından doğrudan işlenebilir hale getirilmiştir (Yin vd., 2021).

Layer (type)	Output Shape	Param #
input (InputLayer)	(None, 3, 1)	0
conv1 (Conv1D)	(None, 3, 64)	256
bn1 (BatchNormalization)	(None, 3, 64)	256
relu1 (ReLU)	(None, 3, 64)	0
conv2 (Conv1D)	(None, 3, 128)	24,704
bn2 (BatchNormalization)	(None, 3, 128)	512
relu2 (ReLU)	(None, 3, 128)	0
gmp (GlobalMaxPooling1D)	(None, 128)	0
fc1 (Dense)	(None, 256)	33,024
bn3 (BatchNormalization)	(None, 256)	1,024
relu3 (ReLU)	(None, 256)	0
drop1 (Dropout)	(None, 256)	0
fc2 (Dense)	(None, 128)	32,896
bn4 (BatchNormalization)	(None, 128)	512
relu4 (ReLU)	(None, 128)	0
drop2 (Dropout)	(None, 128)	0
out (Dense)	(None, 7)	903

Şekil 6. CNN mimarisi

Model iki konvolüsyon katmanından oluşmaktadır. İlk katmanda 64 filtre, ikinci katmanda ise 128 filtre kullanılmıştır. Her iki katmanda da ReLU aktivasyonu uygulanmış, ardından Batch Normalization ile öğrenme süreci stabilize edilmiştir. İkinci konvolüsyon katmanından sonra Global Max Pooling işlemi yapılarak tüm noktaların özellikleri tek bir öznitelik vektöründe özetlenmiştir. Bu öznitelik daha sonra tam bağlantılı bir katmana aktarılmış ve softmax aktivasyon fonksiyonu aracılığıyla yedi sınıfa ayrılmıştır. Bu sınıflar sırasıyla araba, yaya, otobüs, kamyon, motosiklet, bariyer ve bisiklet rafı'dır. Modelin eğitimi sırasında kayıp fonksiyonu olarak categorical cross-entropy kullanılmıştır. Optimizasyon algoritması olarak Adam tercih edilmiş ve öğrenme süreci 40 epoch boyunca sürdürülmüştür. Eğitim ve doğrulama doğruluklarının birbirine yakın seyretmesi, modelin aşırı öğrenme yapmadan genelleme yeteneği kazandığını göstermektedir. CNN modeli, özellikle fazla örneğe sahip araba ve kamyon gibi sınıflarda görece yüksek

performans göstermiştir. Ancak az örneğe sahip bariyer ve bisiklet rafı gibi sınıflarda performansın daha düşük kaldığı gözlenmiştir. Bu durum, sınıf dengesizliğinin doğrudan bir sonucu olarak değerlendirilebilir. Genel doğruluk oranı yaklaşık %83 civarında gerçekleşmiştir (Ioffe & Szegedy, 2015; Johnson & Khoshgoftaar, 2019).

2.4.3. Pointnet-Mini

Çalışmada nokta bulutu sınıflandırmasında kullanılan yöntemlerden biri de PointNet mimarisidir. MLP ya da CNN gibi geleneksel yaklaşımlarda nokta bulutu genellikle voksel ızgaralara dönüştürülmekte veya 2B görüntülere indirgenmektedir. PointNet mimarisi, doğrudan işlenmemiş 3B noktalar üzerinde çalışan ilk derin öğrenme tabanlı modellerden biridir (Qi vd., 2017).

Bu çalışmada PointNet'in mini versiyonu tercih edilmiştir. Google Colab üzerinde sınırlı GPU/TPU kaynaklarının donanım kısıtları ve veri setinin büyüklüğünden dolayı mini versiyon kullanılmıştır. PointNet modelinin tam ölçekli versiyonu milyonlarca parametreye sahip olup daha fazla hesaplama gücü gerektirirken, PointNet mini versiyonu daha az katman ve parametreyle benzer prensipleri koruyarak çalışmayı mümkün kılmıştır.

PointNet, bir nokta bulutundaki her noktayı bağımsız olarak işleyen çok katmanlı algılayıcı katmanları ile başlar. Bu aşamada her nokta için özellikler çıkarılır. Daha sonra, maksimum havuzlama katmanı kullanılarak bütün noktaların özellikleri birleştirilir ve nokta bulutunun küresel temsili elde edilir.

Bu çalışmada PointNet mini modeli diğer yöntemler gibi nuScenes Lidarseg-mini veri seti üzerinde kullanılmıştır. Öncelikle nokta bulutu verisi normalize edilmiş ve her örnek, sabit sayıda nokta seçilerek modele giriş olarak verilmiştir. Modelin giriş boyutu her bir nokta için x, y, z koordinatlarından oluşmaktadır. PointNet Mini'nin katman yapısı Şekil 7'de verilmiştir.

Layer (type)	Output Shape	Param #
input_layer_3 (InputLayer)	(None, 64, 3)	0
conv1d_6 (Conv1D)	(None, 64, 64)	192
batch_normalization_10 (BatchNormalization)	(None, 64, 64)	256
re_lu_14 (ReLU)	(None, 64, 64)	0
conv1d_7 (Conv1D)	(None, 64, 128)	8,192
batch_normalization_11 (BatchNormalization)	(None, 64, 128)	512
re_lu_15 (ReLU)	(None, 64, 128)	0
conv1d_8 (Conv1D)	(None, 64, 256)	32,768
batch_normalization_12 (BatchNormalization)	(None, 64, 256)	1,024
re_lu_16 (ReLU)	(None, 64, 256)	0
global_max_pooling1d_2 (GlobalMaxPooling1D)	(None, 256)	0
dense_11 (Dense)	(None, 256)	65,536
batch_normalization_13 (BatchNormalization)	(None, 256)	1,024
re_lu_17 (ReLU)	(None, 256)	0
dropout_7 (Dropout)	(None, 256)	0
dense_12 (Dense)	(None, 128)	32,768
batch_normalization_14 (BatchNormalization)	(None, 128)	512
re_lu_18 (ReLU)	(None, 128)	0
dropout_8 (Dropout)	(None, 128)	0
dense_13 (Dense)	(None, 7)	983

Şekil 7. Pointnet-Mini katman yapısı

Mini modelinin katman yapısı incelendiğinde, giriş katmanında her noktanın üç boyutlu koordinatları modele doğrudan verilmiştir. Özellik çıkarımı aşamasında sırasıyla 64, 128 ve 256 filtreye sahip üç adet Conv1D katmanı kullanılmıştır. Bu katmanların her birinden sonra, modelin daha kararlı öğrenmesini sağlamak için batch normalization uygulanmış, doğrusal olmayan ilişkileri yakalayabilmek amacıyla ReLU aktivasyon fonksiyonu tercih edilmiştir.

Nokta bazlı öznitelikler çıkarıldıktan sonra, global havuzlama işlemi ile bu öznitelikler bir araya getirilmiş ve nokta bulutunun küresel temsili elde edilmiştir. Bu küresel temsil, sınıflandırma aşamasında kullanılan tam bağlantılı katmanlara aktarılmıştır. İlk olarak 256 nöronlu bir tam bağlantılı katman eklenmiş, bu katmanda ReLU aktivasyonu, batch normalization ve düzenleleştirme amacıyla dropout uygulanmıştır. Bunu takiben 128 nöronlu bir başka tam bağlantılı katman yine aynı yapı ile tasarlanmıştır. Çıkış

katmanında ise 7 nöron yer almakta olup, her biri araba, yaya, otobüs, kamyon, motosiklet, bariyer ve bisiklet rafı sınıflarını temsil etmektedir. Modelin genelinde aşırı öğrenme riskini azaltmak amacıyla tam bağlantılı katmanlarda dropout düzenleme yöntemi kullanılmıştır. Böylece modelin hem sınıflandırma başarısının hem de genelleme kabiliyetinin artırılması hedeflenmiştir. PointNet mini modelinin tercih edilmesinin birkaç önemli nedeni vardır:

- Doğrudan nokta tabanlı yaklaşım: Veri kaybı olmaksızın, 3B nokta bulutunu ham haliyle işleyebilmesi.
- Hız ve verimlilik: Voksel tabanlı CNN yöntemleri daha yüksek hesaplama maliyetine sahipken, PointNet daha az işlem yüküyle daha hızlı sonuç vermektedir.
- Kaynak kısıtları: Colab ortamındaki hesaplama sınırları göz önüne alındığında PointNet mini, uygulanabilir ve pratik bir çözüm olmuştur.

Google Colab üzerinde gerçekleştirilen deneylerde, PointNet mimarisi şu şekilde uygulanmıştır: Her nokta için MLP katmanları kullanılarak nokta bazlı öznitelikler çıkarılmıştır. Nokta bazlı özniteliklerden global öznitelik vektörü elde etmek üzere maksimum havuzlama işlemi uygulanmıştır. Tam bağlantılı katmanlar aracılığıyla sınıflandırma yapılmıştır. Kayıp fonksiyonu olarak Categorical Cross-Entropy seçilmiştir. Optimizasyon için Adam algoritması tercih edilmiştir. PointNet mini modelimizde eğitim sürecinde kayıp değerlerinin düzenli olarak azaldığı, doğruluk oranlarının ise yaklaşık %85 seviyelerine ulaştığı gözlemlenmiştir. Bu değer, MLP tabanlı yöntemlere kıyasla daha yüksek bir genel doğruluk sağlamış ve özellikle araç ve yaya ayrımı gibi geometrik karmaşıklığı yüksek olan sınıflarda daha iyi performans göstermiştir. Ancak tam ölçekli PointNet'in kullanılamaması nedeniyle modelin bazı küçük sınıflarında ayırt etme başarısının sınırlı kaldığı da tespit edilmiştir (Qui vd., 2017).

2.5. Deney Ortamı

Bu çalışmadaki tüm kodlar ve deneyler, Google'ın sunduğu ücretsiz bir platform olan Google Colab üzerinde gerçekleştirilmiştir. Colab, bulut tabanlı bir çalışma ortamı olduğu için bilgisayarda güçlü bir donanıma ihtiyaç duyulmadan Python kodları yazılıp çalıştırılmasına olanak tanımıştır.

Çalışmada Colab ortamının tercih edilmesinin en büyük nedenlerinden biri, Colab'ın GPU desteği sunmasıdır. Bu çalışmada kullanılan veriler Lidar sensörlerinden geldiği için yüksek boyutluydu ve işlenmeleri ciddi hesaplama gücü gerektiriyordu. Colab sayesinde, zaman zaman NVIDIA Tesla T4 tipi bir GPU ve yaklaşık 12 GB RAM ile çalışmak mümkün olmuştur. Bu da hem model eğitimi hızlandırmış hem de kullanılan bilgisayarda oluşabilecek yetersizlikleri ortadan kaldırmıştır.

Çalışma sırasında öncelikle veriler Google Drive'a yüklenerek Colab'a bağlandı. Böylece büyük dosyalar rahatlıkla okunabildi ve eğitim süreçleri sekteye uğramadan ilerlemiştir. Ayrıca Colab, Jupyter Notebook benzeri bir arayüze sahip olduğu için kodları hücre hücre yazmak, çalıştırmak ve yorumlamak oldukça pratikti. Ancak Colab'ın bazı sınırlamaları da vardı. Örneğin, oturumlar genellikle 12 saat içinde otomatik olarak kapanıyordu. Bu nedenle uzun süren model eğitimlerinde bazen yeniden başlatmak gerekebiliyordu. Ayrıca bazı büyük veri işlemlerinde belleğin sınırına ulaşmak da mümkün oluyordu. Bu gibi durumlarla karşılaşıldığında veri parçalara bölünerek işlenmeye çalışıldı veya eğitim döngüleri kısaltıldı.

Sonuç olarak, Google Colab bu tez çalışması için hem erişilebilir hem de güçlü bir ortam sunmuştur. GPU destekli modelleri eğitmek, büyük verileri işlemek ve deneyleri yönetmek açısından oldukça yeterli bir platform olduğu görüldü.

2.6. Kullanılan Kütüphane, Fonksiyon ve Modüller

Bu çalışmada Lidar verilerini işlemek, görselleştirmek ve sınıflandırmak için farklı Python kütüphanelerinden yararlanılmıştır. Bu kütüphanelerden bazıları çok temel gibi görünse de, bir tanesinin eksik olduğu durumda kodlar hata vermiştir. Her bir kütüphane, fonksiyon ve modül farklı bir görevi yerine getirmektedir. Bu bölümde çalışmada kullanılan hem önemli hem küçük yardımcı kütüphaneler ve fonksiyonlar ayrımsız olarak anlatılmıştır ve çalışmadan alınan kısa kod örnekleriyle birlikte açıklanmıştır.

2.6.1. Numpy Kütüphanesi

Çalışma kapsamında sayısal verilerin düzenlenmesi, etiketlerin kontrol edilmesi ve verilerin uygun yapıya getirilmesi gibi işlemler için NumPy kütüphanesi kullanılmıştır. Bu kütüphane ile noktalar normalize edilirken, ortalama ve standart sapma gibi işlemler bu kütüphane ile yapılmıştır (Harris vd., 2020).

- Kullanılan fonksiyonlar:

`np.array()` – Ham verileri, modelde kullanılabilir hale getirmek için Python listeleri NumPy dizilerine dönüştürülmüştür

`np.mean()` – Genel başarıyı özetlemek amacıyla ortalama değer bu fonksiyonla hesaplanmıştır.

`np.unique()` – Etiketlerin hangi sınıflara ait olduğunu belirlemek, kaç farklı sınıf olduğunu anlamak ve sınıf dağılımını kontrol etmek için kullanılmıştır.

`np.concatenate()` – Eğitim verisinin ve test verisinin aynı yapıya getirilmesi sırasında kullanılmıştır.

```
import numpy as np

# deneyle kullandığımız 4 sınıf ve sınıf id leri
class_map = {
    1: 0, # araba → sınıf 0
    4: 1, # yaya → sınıf 1
    5: 2, # otobüs → sınıf 2
    7: 3  # kamyon → sınıf 3
}
```

Şekil 8. Numpy kütüphanesi örnek kod çıktısı

2.6.2. Google.colab Modülü

Google Drive’a yüklenen veri setlerine Colab üzerinden erişmek için `google.colab` modülü kullanılmıştır. Bu modül bir Python kütüphanesi değildir.

```
from google.colab import drive
import os

# Google Drive'a bağlanmak ve dosya yolu seçmek için kullanıldı
drive.mount('/content/drive')
```

Şekil 9. Google.colab modülü örnek kod çıktısı

2.6.3. Matplotlib Kütüphanesi

Bazı durumlarda hızlıca bir histogram veya sınıf dağılım grafiği çizmek istenilen durumlarda matplotlib kütüphanesi ile iki boyutlu grafikler oluşturulmuştur. Özellikle eğitim doğruluklarını çizmekte kullanılmıştır (Hunter, 2007).

- Kullanılan Fonksiyonlar

plt.plot()- Eğitim ve doğrulama sırasında oluşan doğruluk/kayıp değerlerini çizgi grafiği şeklinde göstermek için kullanıldı.

plt.xlabel(), plt.ylabel()- X eksenini genellikle "Epoch" (eğitim turu) olarak adlandırıldı ve Y eksenini ise doğruluk ya da kayıp değerini gösterdi.

plt.legend(), plt.title()- Grafikte hangi eğrinin neyi temsil ettiğini açıklamak için legend kullanıldı. Ayrıca her grafik için uygun bir başlık verildi.

plt.show() - Grafiği ekrana bastırmak ve çıktığı görmek için bu fonksiyon kullanıldı.

```
import matplotlib.pyplot as plt

# Sınıf etiketleri ve karşılık gelen sayılar
class_labels = ['car', 'pedestrian', 'bus', 'truck', 'motorcycle', 'barrier', 'bicycle_rack']
class_counts = [47184, 8448, 1435, 52366, 711, 3631, 591]
```

Şekil 10. Matplotlib kütüphanesi örnek kod çıktısı

2.6.4. Plotly Kütüphanesi

Bu çalışmada sınıflandırma sonuçlarının görsel olarak incelenebilmesi amacıyla 3 boyutlu görselleştirme oldukça önemli olduğu ve matplotlib bu noktada yetersiz kaldığı için, daha etkileşimli olan Plotly kütüphanesinden yararlanılmıştır. Plotly sayesinde sınıflandırılmış Lidar noktaları kolayca 3B olarak renklendirilebildi ve döndürülüp yakınlaştırılabilirdi (Sievert, 2020).

- Kullanılan Fonksiyonlar

plotly.express

px.scatter_3d() – Sınıflandırma sonuçlarını görselleştirmek için kullanılmıştır bu fonksiyon sayesinde sahneyi döndürülebilmiş yakınlaştırılabilmiştir.

plotly.graph_objects - Bazı durumlarda veri üzerinde daha fazla kontrol sağlamak gerektiğinde, plotly.graph_objects modülünün Scatter3d sınıfı kullanılmıştır. Bu yapı ile, renk skalası, eksen sınırları gibi birçok parametre daha detaylı olarak kontrol edilmiştir.

go.Scatter3d() – Bu fonksiyon sayesinde nokta bulutları elle kontrol edilebilmiştir.

go.Figure() – Karmaşık sahneleri görselleştirirken Figure() fonksiyonu sayesinde farklı görsel bileşenler bir araya getirilmiştir.

```
import plotly.express as px
import pandas as pd

# Tahminleri tüm test verisine uygulayalım
test_xyz = X_test[:, :3]
df = pd.DataFrame(test_xyz, columns=["x", "y", "z"])
df["label"] = y_pred

# 3 boyutlu görselleştirme
fig = px.scatter_3d(df, x="x", y="y", z="z", color=df["label"].astype(str),
                    title="🚗 3B Sınıf Tahminleri (0 = pedestrian, 1 = bus)",
                    opacity=0.7)

fig.show()
```

Şekil 11. Plotly kütüphanesi örnek kod çıktısı

```
import plotly.graph_objects as go

# Sınıf renk haritası
color_map = {
    0: 'red',      # araba
    1: 'green',    # yaya
    2: 'blue',     # otobüs
    3: 'yellow',   # kamyon
    4: 'purple',   # motosiklet
    5: 'orange',   # bariyer
    6: 'black',    # bisiklet rafı
}
```

Şekil 12. Plotly kütüphanesi kullanım örneği

2.6.5. Scikit-learn (sklearn) Kütüphanesi

Makine öğrenmesi algoritmaları, veri bölme ve model değerlendirme işlemleri için kullanılan bu kütüphane ile MLP ve Random Forest gibi yöntemler uygulanmış ve başarı oranlarının ölçümü bu kütüphane ve fonksiyonları ile yapılmıştır (Pedregosa vd., 2011).

- Kullanılan fonksiyonlar

`train_test_split()` – Veriyi eğitim ve test olarak ayırmak için kullanılmıştır.

`MLPClassifier()` – Çok katmanlı yapay sinir ağıdır. Projenin ilk aşamasında, derin öğrenmeye geçmeden önce çok katmanlı yapay sinir ağı (MLP) çalışmasında kullanılmıştır.

`RandomForestClassifier()` – Karar ağaçlarına dayalı sınıflandırmadır ve çalışmada nokta bulutu sınıflandırması için kullanılan ilk algoritmalardan biriydi.

`classification_report()` – Sınıfa özel başarı metriklerini raporlamak için kullanılmıştır.

`Doğruluk_score()` – Genel başarıyı yüzdesel olarak görmek için kullanıldı.

`MinMaxScaler()` – Nokta koordinatlarını ve yoğunluk değerini 0–1 aralığına ölçeklemek için kullanılmıştır. MLP ve CNN modellerinde bu işlem başarıyı artırmıştır.

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt

# Sadece x, y, z ve yoğunluk
features = X[:, :4]

# normalize adımı
scaler = MinMaxScaler()
features = scaler.fit_transform(features)
```

Şekil 13. Sklearn kütüphanesi örnek kod çıktısı

`train_test_split (sklearn.model_selection)` - Veriyi %80 eğitim / %20 test olarak ayırmakta kullanılmıştır

`make_pipeline (sklearn.pipeline)` - KNN akışında ölçekleme + sınıflandırıcıyı tek bir yerde birleştirmek için kullanılmıştır.

2.6.6. PyTorch (torch) Kütüphanesi

Bu çalışmada CNN modelini tanımlamak ve eğitmek için PyTorch kütüphanesinden faydalanılmıştır. Bu süreçte özellikle tensör işlemleri, model katmanlarının oluşturulması, aktivasyon fonksiyonları, kayıp hesaplama ve optimizasyon adımlarında bazı fonksiyonlar aktif olarak kullanılmıştır (Paszke vd., 2019).

- Kullanılan Fonksiyonlar

`torch.tensor()`- NumPy dizilerini tensör formatına çevirmek için kullanıldı. Model eğitimi için verilerin float32 gibi uygun PyTorch veri tiplerine dönüştürülmesi gerekiyordu.

`torch.device()` ve `torch.cuda.is_available()`- Modelin eğitiminde varsa GPU kullanılabilmesi için cihaz kontrolü yapıldı. Bu fonksiyonlar sayesinde kod, otomatik olarak GPU destekli çalışacak şekilde ayarlandı.

`nn.Module` – Bu modül PyTorch'ta model tanımlamanın standart yöntemidir.

`nn.Linear()` - Girişten çıkışa tam bağlantılı katmanlar kurmak için kullanıldı. Örneğin, girişteki 4 boyutlu veriyi ara katmanlarda 64, 128 gibi boyutlara taşıyıp çıktıda 7 sınıfa göre sonuç üretildi.

`nn.Dropout()`- Modelin overfitting yapmaması için eğitim sırasında bazı bağlantılar rastgele sıfırlandı.

`F.relu()`- Katmanlar arası geçişlerde doğrusal olmayan aktivasyon sağlamak için kullanılmıştır. Özellikle ara katmanlarda modelin öğrenme yeteneğini artırdı.

`F.cross_entropy()` - Modelin tahmin ettiği sınıf ile gerçek sınıf arasındaki farkı ölçmek için kayıp fonksiyonu olarak kullanılmıştır. Bu değer eğitim boyunca minimize edilmeye çalışılmıştır.

`Optim.Adam()`- Modelin ağırlıklarını güncelleyen optimizasyon algoritması olarak Adam tercih edilmiştir. Hızlı öğrenme ve iyi genel performansı sayesinde eğitim sürecinde istikrarlı sonuçlar vermiştir.

```
import torch
import torch.nn as nn
import torch.optim as optim
import torch.nn.functional as F
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score
import numpy as np

# =====
# 1. VERİ HAZIRLAMA
# =====
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)
```

Şekil 14. PyTorch kütüphanesi örnek kod çıktısı

2.6.7. KNeighborsClassifier

Bu fonksiyon eğitimde model kurmayarak veriyi saklar ve testte en yakın k komşunun sınıfına bakar.

- Kullanılan Parametreler:

`n_neighbors = 3-7-11` = Komşu sayısı; komşu sayısının daha büyük olması (k) → daha pürüzsüz karar sınırları sağlar ancak küçük sınıflarda daha fazla kayba neden olur.

`weights="distance"`- Yakın komşulara daha yüksek ağırlık vererek dengesiz sınıflarda dengeyi koruduğu için kullanıldı.

`leaf_size = 40`: Ağaç yapısının yaprak boyutu; hız/performans ayarını yapmak için kullanıldı (Murphy, 2022).

```
# | KÜTÜPHANELER
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import make_pipeline
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
```

Şekil 15. KNeighborsClassifier örnek kod çıktısı

```
knn = make_pipeline(
    StandardScaler(with_mean=True, with_std=True),
    KNeighborsClassifier(
        n_neighbors=7,
        weights="distance",      # yakın komşulara daha çok ağırlık verir
        algorithm="kd_tree",     # 3B için verimlidir
        leaf_size=40,
        n_jobs=-1
    )
)
```

Şekil 16. KNeighborsClassifier kullanım kod örneği

2.6.8. Yardımcı Kütüphaneler (os, json)

Veri kümesinin yüklenmesi ve dosya kontrol işlemleri sırasında Python'un bazı modüllerinden yararlanılmıştır. Bu kütüphaneler, doğrudan sınıflandırma algoritmalarıyla ilgili olmasa da projenin düzgün çalışması için oldukça önemli olmuştur. Her iki kütüphane de Python'un standart kütüphaneleri arasında yer aldığı için ek kurulum gerektirmemiş ve doğrudan kullanılabilmiştir.

- Os

Veri setinin Colab ortamında olup olmadığını kontrol etmek, eksikse uyarı vermek ve klasör işlemlerini yönetmek için os modülü kullanılmıştır. Özellikle .tgz dosyasının gerçekten Drive'a yüklenip yüklenmediğini kontrol etmek amacıyla kullanılmıştır.

```
from google.colab import drive
import os

# Google Drive'ı bağla
drive.mount('/content/drive')

# Dosya yolu
tgz_path = "/content/drive/MyDrive/projem/v1.0-mini.tgz"

# Dosya var mı kontrol et
if os.path.exists(tgz_path):
    print(f"✅ Dosya bulundu: {tgz_path}")
else:
    print(f"❌ Dosya bulunamadı! Yolu kontrol et: {tgz_path}")
```

Şekil 17. Os yardımcı kütüphane örnek kod çıktısı

- Tarfile

NuScenes veri kümesi .tgz (tar-gz) formatında olduğu için, bu dosyayı Colab üzerinde açmak için bu modülden faydalanılmıştır. Böylece tüm veri klasörü manuel yükleme yapılmadan açılarak doğrudan kullanılabilir hâle getirilmiştir.

```
import os
import tarfile

bz2_path = "/content/nuScenes-lidarseg-mini-v1.0.tar.bz2"
extract_path = "/content/nuscenes" # dosyanın çıkarılacağı klasöre

# .tar.bz2 dosyasını açar ve dosyayı çıkartır
with tarfile.open(bz2_path, "r:bz2") as tar:
    tar.extractall(path=extract_path)

print("✅ lidarseg dosyası başarıyla çıkarıldı.")
```

/tmp/ipython-input-1955425777.py:9: DeprecationWarning:
Python 3.14 will, by default, filter extracted tar archives and reject
✅ lidarseg dosyası başarıyla çıkarıldı.

Şekil 18. Tarfile yardımcı kütüphane örnek kod çıktısı

2.7. Deney Kurulumu

Bu çalışmada yapılan deneyler yüksek tekrarlanabilirlik ve adaletli karşılaştırma doğrultusunda tasarlanmıştır. Tüm eğitim ve test süreçleri Google Colab üzerinde yürütülmüştür. Çalışmanın ortamı hem donanım hem de yazılım bileşenleri açısından sabitlenmiş parametrelerle yapılandırılmıştır. Deneylerde, Google Colab ortamında sağlanan NVIDIA Tesla T4 grafik işlemcisi (16 GB VRAM), 2.3 GHz çift çekirdekli sanal CPU ve 12 GB RAM kullanılmıştır. Depolama, Colab çalışma alanı ve Google Drive entegrasyonu üzerinden sağlanmıştır.

Tüm deneylerde tekrarlanabilirlik oranı olan `random_state=42` olarak sabit değer seçilmiştir. Bunun sebebi 42'nin bilimsel çalışmalarda oldukça sık kullanılan bir standart değer olmasıdır bu sayede kodun yeniden çalıştırılmasında aynı veri bölünmesi, aynı model ağırlıkları başlangıcı ve aynı sonuçlar elde edilebilir. PyTorch tarafında `torch.manual_seed(42)` ve `torch.backends.cudnn.deterministic = True` ayarları uygulanarak, GPU tabanlı işlemlerde öngörülebilir sonuçlar elde edilmesi sağlanmıştır. Bu sayede hem CPU hem de GPU üzerinde çalışan rastgele tüm süreçler kontrol altına alınmıştır (Reimers & Gurevych, 2017).

Tüm deneylerde veri seti, %80 eğitim ve %20 test oranıyla bölünmüştür ancak bazı deneylerde %10 doğrulama alt kümesi ayrılmıştır. Bu alt küme, modelin aşırı öğrenmesini önlemek ve hiperparametre seçimlerini optimize etmek amacıyla kullanılmıştır. Bunun nedeni MLP, CNN gibi derin öğrenme tabanlı yöntemlerde hiperparametre optimizasyonu ve erken durdurma için ayrı bir izleme veri kümesine ihtiyaç duyulmasıdır. Random Forest gibi klasik makine öğrenmesi algoritmalarında doğrulama süreci çapraz doğrulama ile yürütülmüş olup ayrı bir doğrulama kümesi gerekliliği bulunmamıştır.

Erken Durdurma: Eğitim sürecinde modelin doğrulama kaybı veya doğruluk değeri izlenir. Belirli sayıda epoch boyunca iyileşme gözlenmezse eğitim sonlandırılır bu sayede de aşırı öğrenme önlenir.

Hiperparametre Ayarı: Öğrenme oranı, dropout oranı, katman sayısı gibi parametreler doğrulama kümesindeki performansa göre optimize edilir.

Adil Karşılaştırma: Random Forest gibi klasik makine öğrenmesi modelleri çapraz doğrulama ile değerlendirilirken derin öğrenme modellerinde eğitim–doğrulama–test ayırımı yapılır. Bu yüzden bazı deneylerde doğrulama alt kümesi ayrılmış, bazılarında ise doğrudan train–test kullanılmıştır.

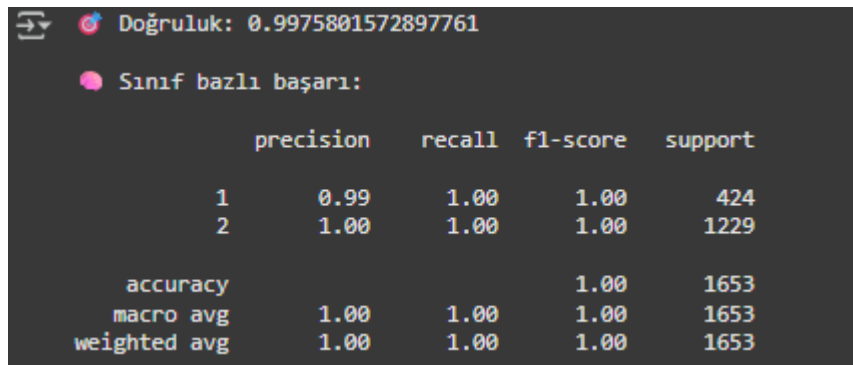
Bölünme, sınıf dağılımını korumak amacıyla stratified sampling yöntemiyle yapılmıştır. Performans değerlendirmesi için doğruluk, macro-F1, macro-Duyarlılık, macro-Kesinlik ve Ortalama IoU (mIoU) metrikleri kullanılmıştır. Sınıf bazlı hata analizleri için karışıklık matrisi görselleri eklenmiş, ayrıca nitel değerlendirme amacıyla tahmin ve gerçek etiketlerin 3B renkli görselleştirmeleri yapılmıştır (Everingham vd., 2010; Pineau vd., 2021).

2.8. Temel Modeller ve Deney Süreçleri

Bu çalışmada kullanılan modeller ve deney süreçlerinde farklı algoritma türlerinin nokta-bazlı Lidar sınıflandırma performanslarını karşılaştırmak amacıyla kademeli olarak ilerlenilmiştir. Deneylerde hem klasik makine öğrenmesi hem de derin öğrenme yaklaşımları kullanılarak algoritma karmaşıklığının ve temsil gücünün model performansına etkisi incelenmiştir.

2.8.1. Random Forest Yöntemi ile Deneysel Sonuçlar

Bu çalışmada Random Forest algoritması ile çalışmanın başlangıç aşaması olarak yöntemin doğruluğunu test etme ve veri işleme adımlarının ne kadar sağlıklı çalıştığını görmek amacıyla düşük sayıda sınıf ile denemeler yapılmıştır. İlk olarak iki sınıf üzerinde bir sınıflandırma gerçekleştirilmiştir ve modelin genel doğruluk oranı %99 seviyesinde elde edilmiş, her iki sınıf için de yüksek kesinlik ve duyarlılık değerleri kaydedilmiştir.



```

→ Doğruluk: 0.9975801572897761

● Sınıf bazlı başarı:

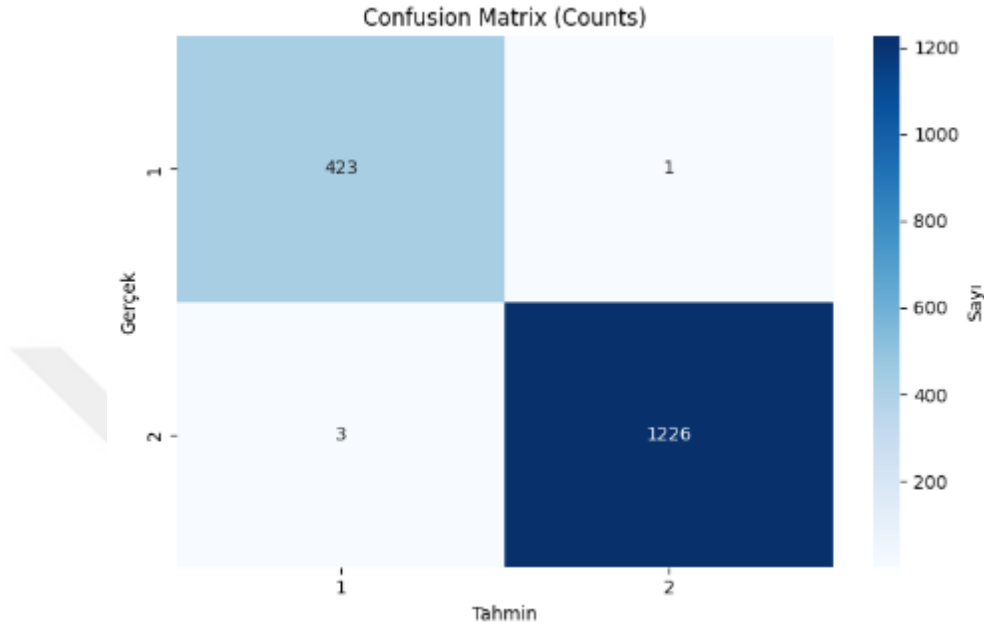
      precision    recall  f1-score   support

     1         0.99         1.00         1.00         424
     2         1.00         1.00         1.00        1229

 accuracy          1.00          1.00          1.00        1653
 macro avg          1.00          1.00          1.00        1653
 weighted avg          1.00          1.00          1.00        1653
  
```

Şekil 19. 2 sınıf için Random Forest sınıf bazlı başarı raporu

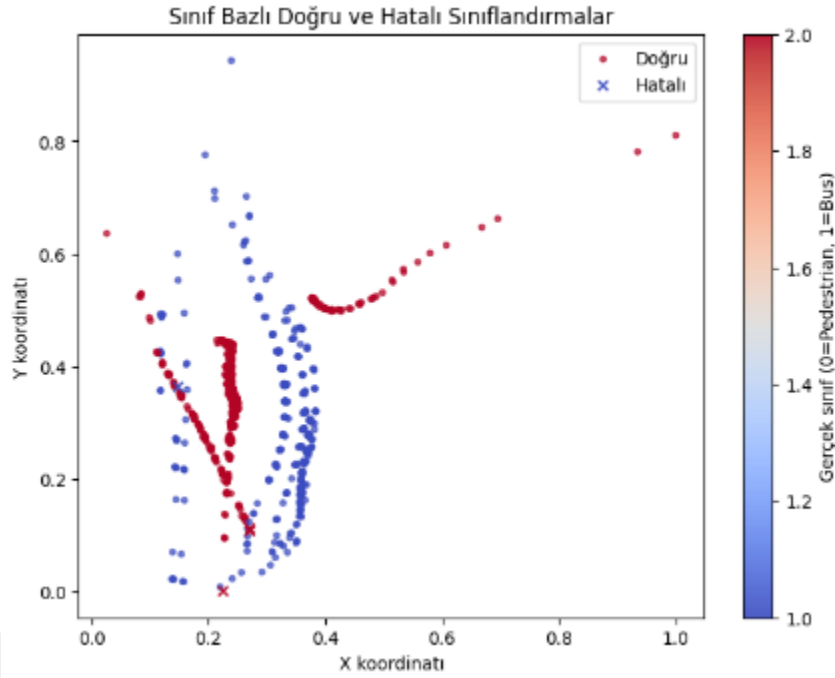
İki sınıflı yaya ve otobüs sınıflandırma deneylerinde elde edilen sonuçlara bakıldığında modelin deneylerde kullanılan test verileri üzerinde hesaplanan doğruluk oranı %99,75 olarak elde edilmiştir.



Şekil 20. Sınıfta sınıflandırma sonucunun karışıklık matrisi

Şekil 20’de satırlar gerçek etiketleri, sütunlar ise modelin tahmin ettiği etiketleri gösterirken diyagonal hücreler doğru sınıflandırmaları, diyagonal dışındaki hücreler ise yanlış sınıflandırmaları gösterir. Şekilde de görüldüğü gibi 423 adet yaya doğru sınıflandırılmış ([0,0] hücresi), 1 adet yaya otobüs olarak tahmin edilmiş ([0,1]), 1226 adet otobüs doğru sınıflandırılmış ([1,1]), 3 adet otobüs yaya olarak tahmin edilmiştir ([1,0]).

Böylece toplam 1653 örneğin yalnızca 4 tanesi hatalı sınıflandırılmıştır. Hesaplanan doğruluk %99 olup kod çıktısı ile uyumludur.



Şekil 21. 2 sınıflı sınıflandırmada doğru-hatalı dağılım

Doğru–hatalı dağılım sınıflandırma grafiği aynı sonucu uzamsal olarak yorumlamayı sağlamaktadır. Grafikte her nokta test kümesindeki bir örneği temsil ediyor olup işaret tipi doğru (•) ve hatalı (×) tahmin ayrımını gösterirken renk skalası ise gerçek sınıfı (0 = yaya, 1 = otobüs) belirtmektedir. Gözlenen birkaç mavi “×” işaretinin çoğunun belirli dar bölgelerde ve az sayıda olmasına karşılık, kırmızı/renkli dolu noktaların yoğun ve yaygın olması, karar sınırının iki sınıfı uzayda büyük ölçüde düzgün ayırdığını göstermektedir. Hatalı noktaların çok az ve dağınık olması, hataların belirli bir kümelenmeye işaret etmediğini ve muhtemelen sınırda kalan veya duyarlılığı düşük ölçümlere ait birkaç örnekten kaynaklandığını düşündürür.

Özetle, iki sınıflı senaryoda yalnızca 1 yaya otobüs sınıfına kaymış, 3 otobüs yaya sınıfına kaymış ve sonuçta 4 örnek yanlış etiketlenmiştir. Bu, hem karışıklık matrisi üzerinden sayısal olarak hem de doğru–hatalı dağılımı üzerinden uzamsal olarak net biçimde görülmektedir ve raporlanan %99,75 doğrulukla tam tutarlıdır. Bu bulgu, yöntemin iki sınıfta güçlü bir ayrım gücüne sahip olduğunu ve daha fazla sınıfa geçildiğinde beklenen zorluk artışını yorumlamak için de iyi bir başlangıç seviyesi sunduğunu göstermektedir.

Genel olarak değerlendirildiğinde, iki sınıflı sınıflandırma deneyleri sonucunda elde edilen bulgular, önerilen yöntemin Lidar nokta bulutu verilerinde yaya ve otobüs ayrımını yüksek doğrulukla gerçekleştirebildiğini göstermiştir.

Daha sonra yaya, araba ve otobüs sınıfları için deneme çalışması yapılmış ve modelin başarısı %99'un üzerinde gözlenmiştir.

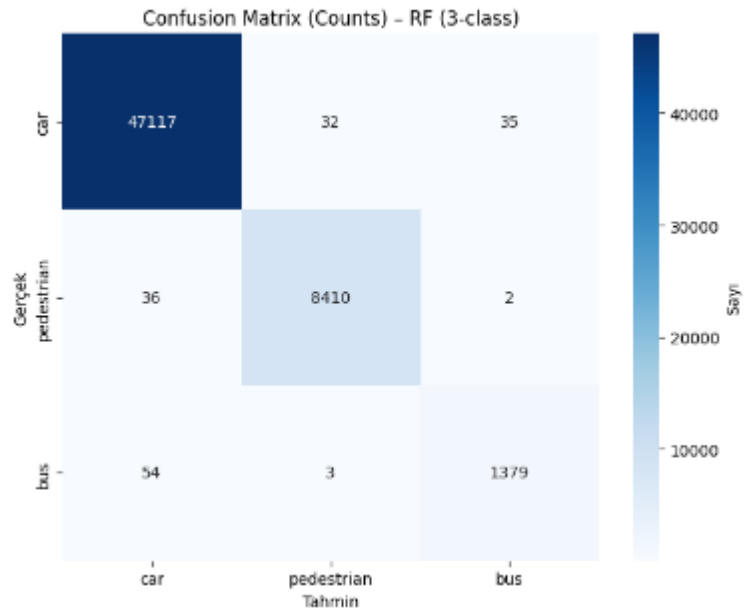
Doğruluk: 0.9972

Sınıf bazlı başarı:

	precision	recall	f1-score	support
car	1.00	1.00	1.00	47184
pedestrian	1.00	1.00	1.00	8448
bus	0.97	0.96	0.97	1436
accuracy			1.00	57068
macro avg	0.99	0.98	0.99	57068
weighted avg	1.00	1.00	1.00	57068

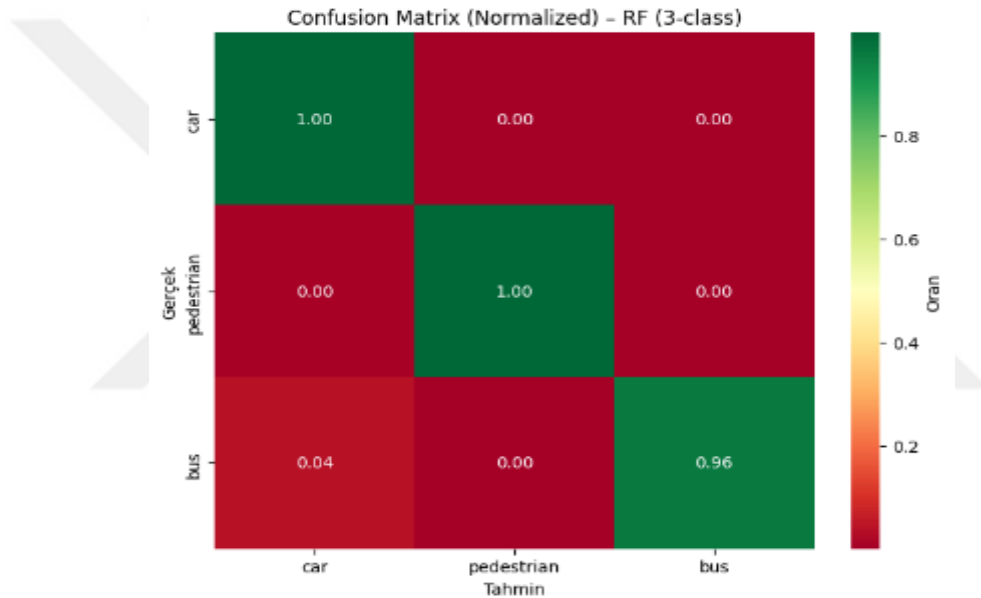
Şekil 22. 3 sınıf için sınıf bazlı başarı raporu

Daha derinlemesine analiz yapabilmek için yalnızca genel doğruluk skoruna değil, sınıf bazlı performansa ve hata dağılımına da bakmak için karmaşıklık matrisi grafiklerinden yararlanılmıştır.



Şekil 23. 3 sınıfta sınıflandırma sonucunun karışıklık matrisi

Karmaşıklık matrisi incelendiğinde, satırların gerçek sınıfları, sütunların ise modelin tahmin ettiği sınıfları temsil ettiği görülmektedir. Köşegen üzerinde yer alan değerler doğru sınıflandırmaları gösterirken, köşegen dışındaki değerler hatalı sınıflandırmaları ifade etmektedir. Sonuçlara göre, otobüs sınıfına ait noktalar model tarafından büyük ölçüde doğru olarak tahmin edilmiştir. Araba sınıfında genel doğruluk yüksek olmakla birlikte, az sayıda noktanın yaya olarak yanlış sınıflandırıldığı gözlenmektedir. Yaya sınıfında ise diğerlerine nazaran daha fazla hata bulunmaktadır ve bazı yayalar araba sınıfına kaydırılmıştır. Bu durum, modelin küçük ve dağınık yapıda olan yaya noktalarını ayırt etmekte zorlandığını göstermektedir.



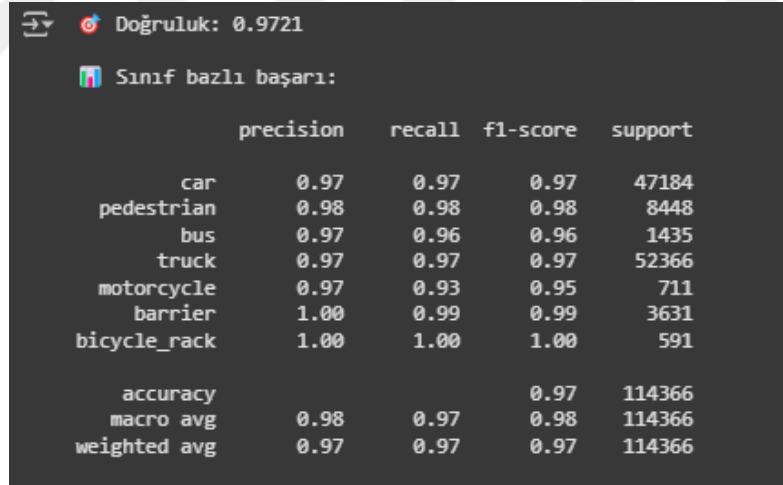
Şekil 24. 3 sınıfta sınıf bazlı metrik raporu

Şekil 24'te sunulan sınıf bazlı metrik raporu modelin performansını daha ayrıntılı bir şekilde göstermektedir. Kesinlikdeğeri, modelin belirli bir sınıf için yaptığı tahminlerin ne kadarının doğru olduğunu, duyarlılık değeri, o sınıfa ait gerçek noktaların yüzde kaçının doğru tahmin edildiğini, F1-score ise bu iki ölçütün birleşimini ifade etmektedir. Çıktı incelendiğinde, otobüs sınıfı için kesinlikve duyarlılık değerlerinin oldukça yüksek olduğu ve bu nedenle modelin bu sınıfta güçlü bir performans sergilediği görülmektedir. Araba sınıfında kesinlik yüksek olmakla birlikte duyarlılık değerinde küçük bir düşüş vardır; yani model bazı arabaları farklı sınıflara kaydırmıştır. Yaya sınıfında ise duyarlılık değerinin diğerlerine kıyasla daha düşük olduğu gözlenmektedir, bu da modelin yayaların tamamını yakalayamadığını, bir kısmını farklı sınıflar altında değerlendirdiğini göstermektedir.

Bu çıktılar birlikte değerlendirildiğinde, modelin özellikle büyük nesnelerde oldukça başarılı olduğu, ancak daha küçük ve yaya gibi karmaşık yapılarda hata oranının arttığı sonucuna varılabilir. Bu sonuçlar, kullanılan Random Forest algoritmasının düşük sınıf sayısında oldukça başarılı çalıştığını ve veri etiketleme sürecinin doğru ilerlediğini göstermiştir.

Bu aşamada temel olarak Doğruluk ve sınıf bazlı kesinlik-duyarlılık-F1 skorları raporlanmıştır. IoU ve mIoU metrikleri, düşük sayıda sınıflar kullanılması nedeniyle bu kısımda detaylandırılmamış asıl değerlendirmeler daha geniş sınıf kümesi olan 7 sınıf üzerinde yapılmıştır (Everingham vd., 2010).

Random Forest algoritması 2 ve 3 sınıf ile deneme çalışmasının ardından, Lidar nokta bulutlarının yedi farklı sınıfa ayrılması amacıyla kullanılmıştır. Model öncelikle $n_estimators = 100$ ve $max_depth = 25$ hiper parametreleri ile yapılandırılmış ve $random_state = 42$ tekrar değeriyle sabitlenerek kararlı sonuçlar elde edilmesini sağlanılmıştır. Eğitim ve test veri kümeleri, %80 eğitim – %20 test oranında ve sınıf dağılımı korunarak ayrılmıştır.



```

Doğruluk: 0.9721

Sınıf bazlı başarı:

      precision    recall  f1-score   support

   car           0.97       0.97       0.97     47184
 pedestrian       0.98       0.98       0.98      8448
    bus           0.97       0.96       0.96      1435
   truck           0.97       0.97       0.97     52366
 motorcycle       0.97       0.93       0.95       711
   barrier         1.00       0.99       0.99      3631
 bicycle_rack     1.00       1.00       1.00       591

 accuracy                   0.97    114366
 macro avg           0.98       0.97       0.98    114366
 weighted avg        0.97       0.97       0.97    114366

```

Şekil 25. 7 sınıf için sınıf bazlı başarı raporu

Kod çıktısında sınıf bazlı değerlendirmeye göre; Araba, kamyon sınıfında %97 kesinlik ve duyarlılık değerleri ile dengeli ve yüksek performans göstermiştir bunun nedeni bu sınıfların veri kümesinde yoğun şekilde bulunmasından dolayı modelin ayırt etmesinin kolay olmasıdır.

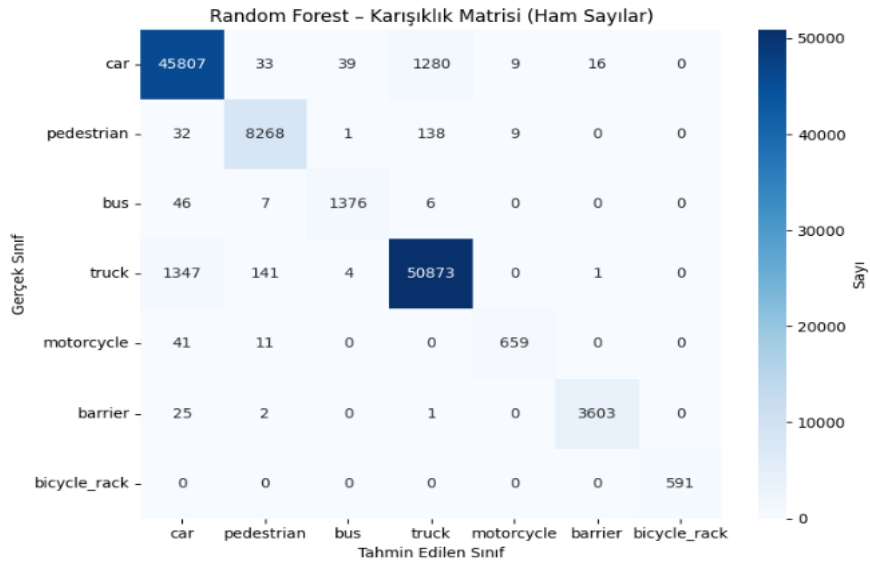
Yaya sınıfında %98 kesinlik/duyarlılık ile modelin küçük ve ince yapıları nesneleri bile yüksek doğrulukla yakalayabildiğini göstermektedir.

Otobüs sınıfında kesinlik %97, duyarlılık %96 ile güçlü, ancak duyarlılık değerinin diğer büyük araç sınıflarına göre daha düşük olması, bazı otobüs noktalarının kamyon veya araba sınıfına karışmasından kaynaklı olabileceği varsayılmaktadır.

Motosiklet sınıfında kesinlik %97, duyarlılık %93 ile diğer sınıflara kıyasla düşük duyarlılık değeri görülmektedir. Bunun sebebi, motosikletlerin daha az sayıda örneğe sahip olması ve 3B noktaların geometrik benzerlik nedeniyle diğer küçük nesnelerle karışabilmesidir.

Bariyer sınıfında kesinlik %100, duyarlılık %99 ile neredeyse kusursuz sınıflandırma yapılmıştır. Noktaların geometrik formu ve sabit konumda olması ayırt ediciliği artırmıştır.

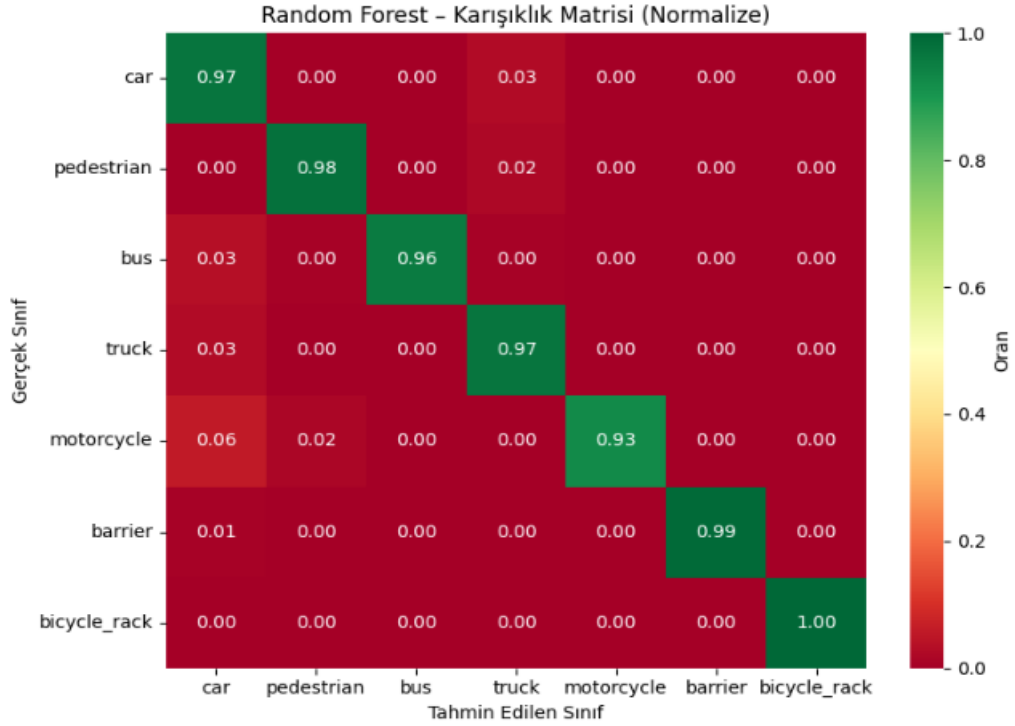
Bisiklet rafı sınıfında kesinlik ve duyarlılık değerlerinin her ikisi de %100'dür. Veri kümesinde sınıfın 591 gibi az sayıda nokta bulundurmasına rağmen modelin bu nesneleri hatasız sınıflandırabilmesi, belirgin geometrik formun etkisinden kaynaklanabilir. Makro ortalama değerleri %98 kesinlik, %97 duyarlılık ile modelin tüm sınıflarda dengeli performans sergilediğini, Ağırlıklı ortalama değerleri ise %97 kesinlik ve duyarlılık ile sınıf dağılımındaki dengesizliklere rağmen genel başarımın korunduğunu göstermektedir. Modelin değerlendirme sonuçları karışıklık matrisleri ile Şekil 26 ve Şekil 27'de sunulmaktadır.



Şekil 26. 7 sınıf için karışıklık matrisi

Şekil 26 modelin test veri kümesindeki her sınıf için tahmin performansını ham sayı cinsinden göstermektedir. Bu görselde yatay eksen modelin tahmin ettiği sınıfları gösterirken dikey eksen veri kümesinde yer alan gerçek sınıfları ifade etmektedir.

Köşegen hücreler ise doğru sınıflandırılan örneklerin sayısını gösterir. Örneğin, araba sınıfındaki 45.807 örnek doğru tahmin edilmiştir. Köşegen dışında kalan hücreler yanlış sınıflandırmaları temsil etmektedir. Örneğin, kamyon sınıfına ait 1.347 örnek yanlışlıkla araba olarak tahmin edilmiştir. Bu görsel ile hangi sınıflar arasında karışıklık yaşandığı görülmektedir. Örneğin araba ↔ kamyon arasında karşılıklı yanlış sınıflandırmalar vardır. Arabanın 1.280 örneği kamyon olarak, kamyonun 1.347 örneği araba olarak tahmin edilmiştir. Bariyer ve bisiklet rafı sınıflarında ise hata oranı oldukça düşüktür; neredeyse tüm örnekler doğru tahmin edilmiştir.



Şekil 27. 7 sınıf için normalize edilmiş karışıklık matrisi

Şekil 27, aynı karışıklık matrisinin 0 ile 1 arasında normalize edilmiş oranlarla gösterilmiş halidir. Bu sayede, sınıf büyüklüklerinden bağımsız olarak hata oranlarının karşılaştırılması mümkün olmaktadır. 1.0 değeri (yeşil tonlar) ilgili sınıftaki tüm örneklerin doğru sınıflandırıldığını, 0.0 değeri ise hiç doğru tahmin yapılamadığını ifade etmektedir. Ara değerler ise ilgili sınıfa ait örneklerin %97'sinin doğru tahmin edildiğini gösterir. Normalize edilmiş matrisin avantajı, farklı büyüklükteki sınıfların hata oranlarının adil

şekilde karşılaştırılabilmesidir. Örneğin Motosiklet sınıfında doğruluk oranı 0.93 ile diğer sınıflara göre daha düşüktür, bu da modelin bu sınıfı ayırt etmekte biraz daha zorlandığını gösterir. Bisiklet rafı sınıfı ise 1.00 ile tamamen doğru sınıflandırılmıştır. Bu sonuçlar, Random Forest algoritmasının Lidar verilerinde farklı nesne sınıflarını yüksek başarıyla ayırt edebildiğini göstermektedir. Ancak, bazı sınıflarda meydana gelen düşük oranlı karışmaların, özellikle sınıflar arası fiziksel benzerliklerin ve sensör kaynaklı veri hatalarının etkisiyle ortaya çıktığı değerlendirilmektedir.

Modelin test kümesindeki genel doğruluk oranı %97,21 olarak elde edilmiştir. Bu metrik, tüm sınıflardaki doğru tahminlerin toplam örnek sayısına oranıdır. Ancak doğruluk tek başına modelin her sınıfta ne kadar dengeli performans gösterdiğini yansıtmayacağı için sınıf-başlı kesişim-birleşim oranı (IoU) ve ortalama IoU (mIoU) değerleri de hesaplanmıştır. IoU, her bir sınıf için doğru tahmin edilen nokta sayısının, o sınıfa ait gerçek ve tahmin edilen tüm noktaların birleşimine oranıdır.

```

Sınıf-başlı IoU:
- car: 0.9411
- pedestrian: 0.9567
- bus: 0.9304
- truck: 0.9458
- motorcycle: 0.9040
- barrier: 0.9877
- bicycle_rack: 1.0000

mIoU: 0.9522

```

Şekil 28. Sınıf bazlı IoU başarı raporu

Yapılan değerlendirmede sınıf-başlı IoU değerleri şu şekilde bulunmuştur:

- Araba: 0.9411
- Yaya: 0.9567
- Otobüs: 0.9304
- Kamyon: 0.9458
- Motosiklet: 0.9040
- Bariyer: 0.9877
- Bisiklet rafı: 1.0000

Bu değerlerin ortalaması olan mIoU, 0.9522 olarak hesaplanmıştır. Buradan anlaşılacağı üzere IoU metriği, genel doğruluk olan %97,21 değerine kıyasla biraz daha düşük çıkmıştır. Bunun temel nedeni, IoU'nun yanlış pozitif ve yanlış negatif tahminleri daha sert bir şekilde cezalandırmasıdır. Örneğin, araba ve kamyon sınıfları arasında yaşanan karşılıklı karışmalar bu sınıfların IoU değerlerini doğruluk oranına göre daha fazla düşürmüştür.

Sonuç olarak, Random Forest modeli, yüksek genel doğruluk ve mIoU değerleri ile güçlü bir performans sergilemiştir. Ancak büyük araç sınıfları arasındaki (araba–kamyon) karışmalar, mIoU değerlerinde sınırlı bir düşüşe neden olmuştur. Bu durum, Lidar sensörlerinin çözünürlüğü, görüş açısı ve veri setindeki örnek çeşitliliği ile ilişkili olabilmektedir.

2.8.2. DL-MLP Yöntemi ile Deneysel Sonuçlar

Bu çalışmada nokta bulutu sınıflandırması için kullanılan derin öğrenme yöntemlerinden biri de Derin Çok Katmanlı Algılayıcı (DL-MLP) modelidir. MLP'nin temel prensibine dayanan bu yöntemde daha fazla gizli katman ve düzenleme teknikleri kullanılarak klasik MLP'den ayrılmaktadır. Modelin giriş katmanında her nokta için x, y, z koordinatları ile yoğunluk değerlerini girdi olarak alınmıştır. Ardışık tam bağlı katmanlarda ReLU aktivasyonu tercih edilmiş olup aşırı öğrenmeyi engellemek amacıyla dropout katmanları eklenmiştir. Çıkış katmanı ise 7 farklı sınıfa ait olasılıkları üretmek için softmax aktivasyon fonksiyonu ile tasarlanmıştır.

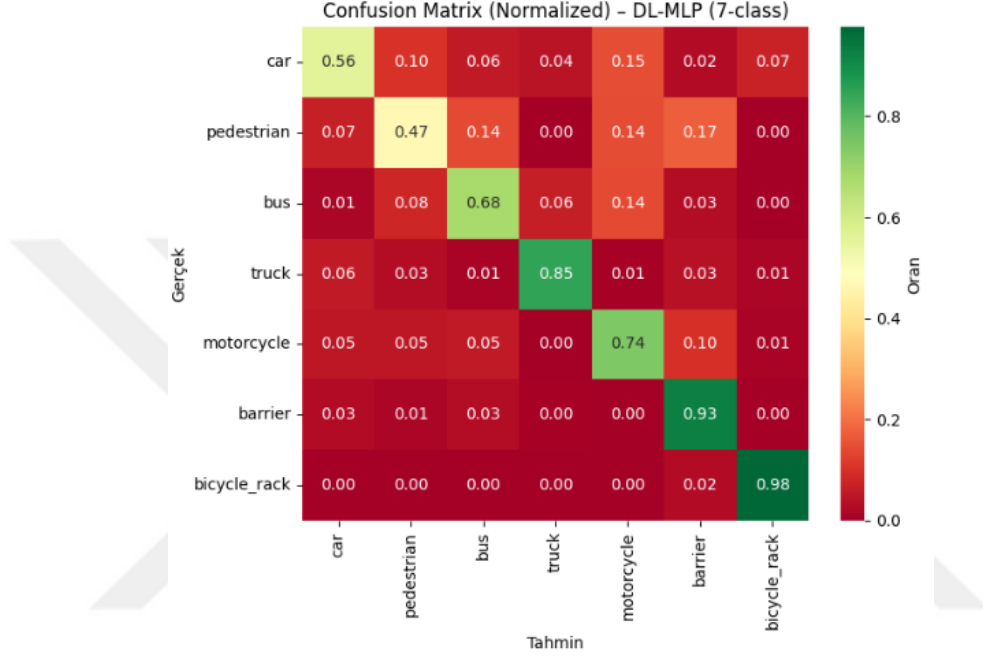
Test Doğruluk: 0.7017

Sınıf Bazlı Rapor:

	precision	recall	f1-score	support
car	0.8786	0.5594	0.6836	47184
pedestrian	0.3775	0.4689	0.4182	8448
bus	0.1748	0.6760	0.2777	1435
truck	0.9558	0.8487	0.8991	52366
motorcycle	0.0574	0.7426	0.1065	711
barrier	0.4258	0.9292	0.5840	3631
bicycle_rack	0.1240	0.9763	0.2200	591
accuracy			0.7017	114366
macro avg	0.4277	0.7430	0.4556	114366
weighted avg	0.8447	0.7017	0.7484	114366

Şekil 29. DL-MLP sınıflandırma bazlı doğruluk raporu

DP-MLP kod çıktısında sınıflara göre elde edilen sınıflandırma metrikleri sunulmaktadır. Araba ve kamyon sınıflarında hem kesinlik hem de duyarlılık değerlerinin yüksek olduğu görülmektedir. Buna karşın, örnek sayısı diğerlerine göre daha az olan motosiklet ve bariyer sınıflarında başarı oranları düşmüştür. Bu durum, veri dengesizliğinin DL-MLP modelinin performansını sınırladığını ortaya koymaktadır.



Şekil 30. 7 sınıf için normalize karışıklık matrisi

Şekil 30’da satırlar gerçek sınıfları, sütunlar ise modelin tahmin ettiği sınıfları göstermektedir. Renk ölçeği 0–1 aralığındadır ve her satır 1’e normalize edilmiştir; dolayısıyla bir satırdaki hücreler, o sınıfa ait örneklerin ne kadarının hangi etikete gittiğini oran olarak vermektedir. Diyagonal hücreler (sol–üstten sağ–alta) sınıf-başı doğru sınıflandırma oranlarını gösterir bu değerler doğrudan duyarlılık ile aynıdır. Diyagonal dışındaki hücreler ise yanlış sınıflandırma örüntülerini ortaya koymaktadır. Sınıf başı bulgulara bakıldığında;

Araba sınıfı: Diyagonal hücre 0.56 olup, gerçek araba noktalarının %56’sı doğru etiketlenmiştir. Kalan önemli paylar motosiklet, yaya, bisiklet rafı ve otobüs etiketlerine dağılmıştır. Bu durum, DL-MLP’nin araba sınıfında duyarlılık açısından zorlandığını ve özellikle motosiklet sınıfına aşırı atama eğilimi bulunduğunu gösterir.

Yaya sınıfı: Doğru sınıflandırma oranı 0.47'dir. Yanlış atamalar ağırlıkla bariyer motosiklet ve otobüs etiketlerine gitmektedir. Yaya noktalarının seyrek/ince yapısı ve sınıflar arası geometrik benzerlikler bu kaymaları açıklamaktadır.

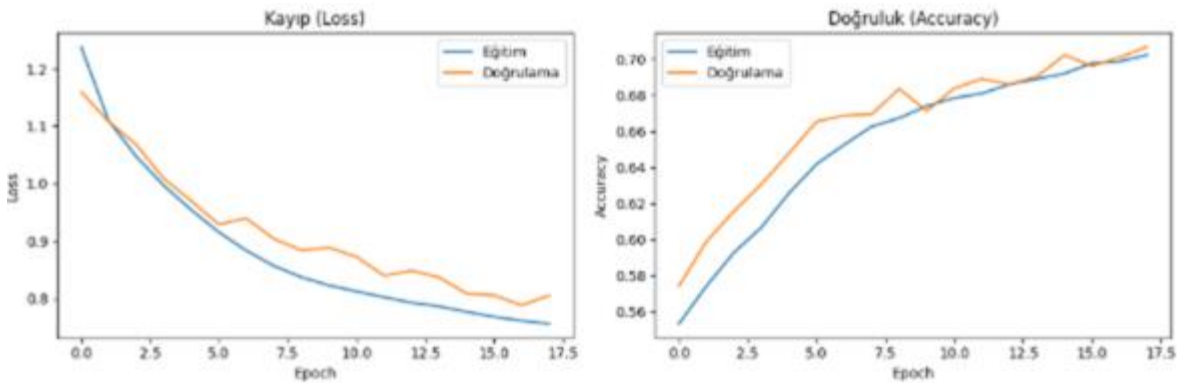
Otobüs sınıfı: Diyagonal oran 0.68 ile orta-yüksektir. Belirgin karışmalar motosiklet ve kamyon yönündedir. Otobüs–kamyon geometrik yakınlığı ve nokta dağılımı buna zemin hazırlamaktadır.

Kamyon Sınıfı: 0.85 ile en yüksek diyagonal oranlardan birine sahiptir; model kamyonları büyük ölçüde doğru yakalamaktadır. Araba gibi küçük oranlı kaymalar az olmak üzere diğer sınıflara görülmektedir.

Motosiklet Sınıfı: Doğruluk oran 0.74 'tür. Ancak diğer sınıfların satırlarında motosiklet sütununa gelen hatalı atamalar da fazladır. Örneğin araba satırında %15, yaya satırında %14 hatalı atama vardır. Bu yorumlandığında motosiklet için duyarlılık yüksek – kesinlik düşük dengesinin bir örneği ortaya çıkmaktadır ve bu sınıf raporundaki çok düşük kesinlik değeri ile tutarlıdır.

Bariyer Sınıfı: Diyagonal 0.93 doğruluk oranı vardır ve model gerçek bariyerleri büyük ölçüde doğru ayırt etmiştir. Ancak özellikle yaya satırında %17'lik bir payın bariyer sütununa gelmesi, bariyer sınıfı için yanlış-pozitif riskinin varlığına işaret etmektedir ve bu kesinliğin sınıf raporunda görece düşük çıkmasıyla uyumludur.

Bisiklet rafı Sınıfı: 0.98 ile matrisin en yüksek doğruluklarından biri olmuştur yalnızca bariyer sütununa %2 düzeyinde bir kayma vardır. Bu sınıfta duyarlılık çok yüksektir ancak diğer sınıflardan özellikle de araba satırındaki %7 kaymanın bu sütuna gelen etiketlerinin kesinliği düşürmektedir. Bu da sınıf raporunda bu sınıf için kesinlik çok düşük duyarlılık değerinin daha yüksek olması ile uyumludur.



Şekil 31. DP-MLP'nin eğitim/doğrulama kaybı ve doğruluk eğrileri

Şeklin sol panelinde kayıp sağ panelinde doğruluk değerlerinin yatay eksene göre değişimi gösterilmektedir. Mavi eğriler eğitim, turuncu eğriler doğrulama kümesine aittir. Eğitim kaybı ilk epochta yaklaşık 1.25 civarından başlayıp düzenli biçimde yaklaşık 0.75 seviyesine kadar düşmektedir. Bu eğilim, her yinelemede veriden daha tutarlı örüntüler öğrendiğini ve optimizasyonun istikrarlı ilerlediğini göstermektedir. Doğrulama kaybı ise yaklaşık 1,12 başlayıp genel olarak aşağı yönlü bir iz izlemekte olup 5–12. epoch aralığında küçük dalgalanmalar görülmektedir.

Bu dalgalanmalar iki nedenle beklenmektedir. İlk neden Dropout ve sınıf ağırlıkları gibi düzenlileştirme mekanizmaları mini-batch'ler arası değişkenlik yaratmasından kaynaklanabilirken diğer sebep ise bazı az örnekli sınıfların üzerinde hatanın düzensiz dağılmasının doğrulama kaybına kısa süreli iniş-çıkışlar olarak yansımından kaynaklı olabilir. Ancak doğrulama kaybının genel eğilimi aşağı olduğu için belirgin bir aşırı uyum gözükmemektedir. Eğriler yaklaşık olarak 10–15. epoch arasında doyuma yaklaşmakta, bu da erken durdurma ve öğrenme oranı azaltma mekanizmalarının doğru noktada devreye girdiğini düşündürmektedir. Resmin sağ kısmında bulunan doğruluk paneli incelendiğinde eğitim doğruluğu yaklaşık %55 başlayarak artış eğilimiyle %70 bandına yerleştiği görülmektedir. Doğrulama doğruluğu, ilk birkaç epochtan itibaren eğitim eğrisinin çok yakınında seyrederek ve zaman zaman bir miktar daha yüksek gerçekleşir. Dropout/LayerNorm gibi düzenlileştirme katmanları eğitim sırasında gürültü ekleyerek eğitim doğruluğunu bir miktar baskılar ve doğrulama aşamasında bu gürültü devre dışı kaldığından ölçülen doğruluk bir miktar yükselir. İki eğri arasındaki farkın az olması modelin genelleme boşluğunun küçük olduğuna ve aşırı öğrenme göstermediğine işaret eder.

Bu çıktılar birlikte incelendiğinde DL-MLP'nin istikrarlı biçimde öğrendiği, düzenlileştirmenin eğitim/doğrulama eğrilerini yakın tuttuğu ve eğitim sonunda genel doğruluğun %70,17 düzeyine ulaştığı görülmektedir.

Sonuç olarak DL-MLP yöntemi, basit MLP'ye kıyasla nokta bulutundaki daha karmaşık ilişkileri öğrenebilmiş ve sınıflandırma performansını belirgin şekilde artırmıştır. Ancak %70,17 seviyesinde kalan doğruluk oranı, nokta bulutlarının yüksek boyutlu yapısı ve sınıflar arasındaki benzerlikler göz önüne alındığında, bu yöntemin tek başına yeterli olmadığını göstermektedir.

2.8.3. ML-MLP Yöntemi ile Deneysel Sonuçlar

Çalışmada MLP iki farklı yöntem ile denenmiştir bu kısımda makine öğrenmesi ile 200 iterasyonla eğitilen MLP modelinin sonuçları, genel doğruluk, sınıf bazlı başarı metrikleri ve görselleştirmeler üzerinden detaylı olarak incelenmiştir. Modelin test doğruluğu %90,34 olarak hesaplanmıştır. Ancak sınıf bazlı metrikler incelendiğinde, bazı sınıflarda iyileşmeler gözlenirken bazı sınıflarda dengesizlikler ortaya çıktığı görülmektedir.

```

Toplam sınıf dağılımı: {0: 235920, 1: 42239, 2: 7178, 3: 261828, 4: 3554, 5: 18156, 6: 2955}
Train dağılımı: {0: 188736, 1: 33791, 2: 5743, 3: 209462, 4: 2843, 5: 14525, 6: 2364}
Test dağılımı: {0: 47184, 1: 8448, 2: 1435, 3: 52366, 4: 711, 5: 3631, 6: 591}
Train (oversampled) dağılımı: {0: 209462, 1: 209462, 2: 209462, 3: 209462, 4: 209462, 5: 209462, 6: 209462}

Test Doğruluk: 0.9034

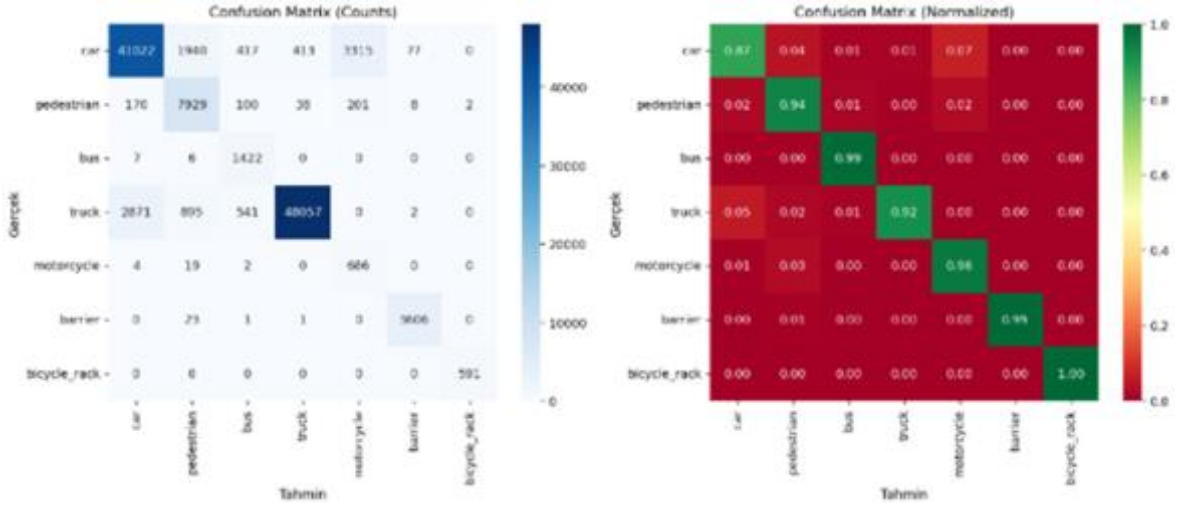
Sınıf bazlı rapor:

```

	precision	recall	f1-score	support
car	0.93	0.87	0.90	47184
pedestrian	0.73	0.94	0.82	8448
bus	0.57	0.99	0.73	1435
truck	0.99	0.92	0.95	52366
motorcycle	0.16	0.96	0.28	711
barrier	0.98	0.99	0.98	3631
bicycle_rack	1.00	1.00	1.00	591
accuracy			0.90	114366
macro avg	0.77	0.95	0.81	114366
weighted avg	0.94	0.90	0.92	114366

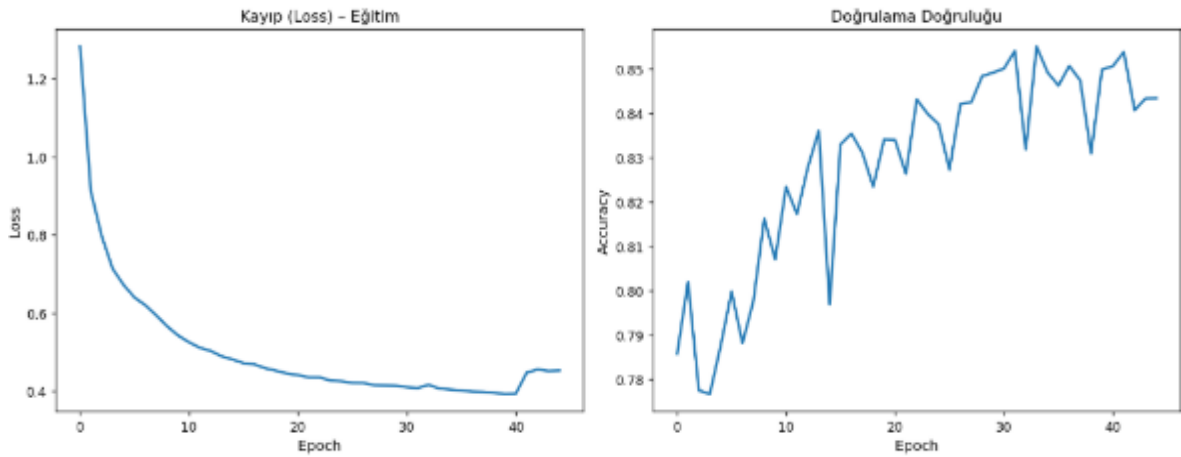
Şekil 32. ML-MLP 7 sınıf için performans raporu

Sınıf bazlı kesinlik, duyarlılık ve F1-skor değerleri incelendiğinde, araba ve kamyon sınıflarında diğer sınıflara oranla daha yüksek başarı elde edilmiştir. Yaya sınıfında duyarlılık değerinin oldukça yüksek olması modelin bu sınıfı öğrenme konusunda başarılı olduğunu, ancak kesinlik değerinin daha düşük olması ise yanlış pozitiflerin fazla olduğunu göstermektedir. Otobüs sınıfında duyarlılık değerinin %99'a yükselmesine rağmen kesinlik %57 seviyesinde kaldığı görülmektedir. Bu durum modelin otobüs sınıfını aşırı tahmin etme eğilimine işaret etmektedir. Düşük örnek sayısına sahip motosiklet sınıfında kesinlik %16 gibi düşük bir seviyede kalmış, buna karşın duyarlılık %96 çıkmıştır. Bu dengesizlik, küçük sınıflarda aşırı örnekleme etkisiyle modelin duyarlılığının artmasına rağmen kesinliğin zarar gördüğünü göstermektedir. Bariyer ve bisiklet rafı sınıflarında ise hem kesinlik hem de duyarlılık değerleri %98–100 aralığında hesaplanmış, bu da bu sınıflarda başarılı bir ayırım yapıldığını göstermektedir.



Şekil 33. 7 sınıf için ML-MLP karışıklık matrisi

Karışıklık matrisi incelendiğinde, araba sınıfındaki örneklerin büyük çoğunluğunun doğru tahmin edildiği, ancak belirli bir kısmının kamyon ve yaya sınıflarıyla karıştığı görülmektedir. Otobüs sınıfında hataların özellikle araba ve kamyon sınıflarına kaydığı gözlenmiştir. Bu durum, görsel olarak birbirine yakın olan veya benzer konum/geometri bilgisine sahip sınıfların model tarafından ayırt edilmesinde zorlanıldığını göstermektedir.



Şekil 34. ML-MLP eğitim ve doğrulama doğruluğu grafiği

Şekil 34'te ML-MLP modelinin eğitim sürecine ait eğitim kaybı ve doğrulama doğruluğu grafikleri verilmiştir. Grafiklerden de görüleceği gibi eğitim kaybı başlangıçta oldukça yüksek bir değerden başlamış olup birkaç iterasyon sonrasında hızlı bir düşüş göstermiştir. Yaklaşık 15. iterasyondan itibaren kayıp değerlerindeki azalma yavaşlamış, 40. iterasyona kadar giderek daha stabil bir seyir izlemiştir. Bu durum, modelin öğrenme

sürecinde erken aşamalarda hızlı bir uyum sağladığını ancak ilerleyen iterasyonlarda hata değerinin giderek daha zor iyileştiğini göstermektedir. Doğrulama doğruluğu grafiği incelendiğinde, modelin doğruluk değerinin ilk 10 iterasyonda hızlı bir şekilde yükselerek %80 seviyesini aştığı, sonraki iterasyonlarda ise daha yavaş fakat istikrarlı bir artış gösterdiği görülmektedir. Yaklaşık 30–35. iterasyon sonrasında doğruluk değerinin %85 civarına yükseldiği ve daha fazla doğruluk artışının sağlanamadığı görülmektedir. Eğitim süreci, scikit-learn kütüphanesinde iterasyon 200 olarak ayarlanmış olmasına rağmen, modelin 40. iterasyonda durduğu görülmektedir. Bunun nedeni, çalışmada kullanılan erken durdurma stratejisinden kaynaklanmaktadır. Erken durdurma yöntemi doğrulama kaybında 10 ardışık iterasyon boyunca iyileşme gözlemlenmediğinde eğitimi otomatik olarak sonlandırmaktadır. Bundan kaynaklı 40. iterasyondan sonra modelin tekrar öğrenme sağlayamayacağı belirlenmiş ve eğitim bu noktada durdurulmuştur.

Sonuç olarak, grafikte görülen kayıp değerlerinin azalması ve doğruluk eğrisinin 85'te kalması modelin 40. iterasyon sonunda optimum öğrenme noktasına ulaştığını göstermektedir. Bu sayede hem gereksiz hesaplama maliyetinden kaçınılmış hem de aşırı öğrenme riski azaltılmıştır.

```

Toplam sınıf dağılımı: {0: 235928, 1: 42239, 2: 7178, 3: 261828, 4: 3554, 5: 18156, 6: 2955}
Train dağılımı: {0: 188736, 1: 33791, 2: 5743, 3: 209462, 4: 2843, 5: 14525, 6: 2364}
Test dağılımı: {0: 47184, 1: 8448, 2: 1435, 3: 52366, 4: 711, 5: 3631, 6: 591}
Train (oversampled) dağılımı: {0: 209462, 1: 209462, 2: 209462, 3: 209462, 4: 209462, 5: 209462, 6: 209462}

Test Doğruluk: 0.9034

Sınıf bazlı rapor:

```

	precision	recall	f1-score	support
car	0.93	0.87	0.90	47184
pedestrian	0.73	0.94	0.82	8448
bus	0.57	0.99	0.73	1435
truck	0.99	0.92	0.95	52366
motorcycle	0.16	0.96	0.28	711
barrier	0.98	0.99	0.98	3631
bicycle_rack	1.00	1.00	1.00	591
accuracy			0.90	114366
macro avg	0.77	0.95	0.81	114366
weighted avg	0.94	0.90	0.92	114366

Şekil 35. 100 iterasyon ile bazlı performans raporu

Çalışmada iterasyon değeri 100 ve 200 olarak denense de model pratikte yaklaşık 40. iterasyonda durmuştur. Bunun nedeni 40. iterasyondan sonra doğrulama kaybında anlamlı bir azalma gerçekleşmemesidir. Bu nedenle 100 ve 200 iterasyon deneyleri aynı test sonuçlarını üretmiş olup aradaki fark yalnızca üst sınır parametresinde kalmıştır.

2.8.4. KNN Yöntemi ile Deneysel Sonuçlar

Bu çalışmada KNN'nin komşu sayısı k için 3, 7 ve 11 değerleri ile test yapılmıştır. k komşu sayısını azalttıkça eğitim/test doğruluğunun arttığı ancak gürültüye duyarlılık da yükseldiği görülmektedir. Sonuçlar incelendiğinde $k=3$ değeri %96,36 ile en yüksek genel doğruluğu vermiştir.

```
Veri boyutu: (571830, 3) (571830,)
```

```
KNN eğitimi (fit) başlıyor...
```

```
✓ Doğruluk: 0.9636
```

```
Sınıf Bazlı Rapor:
```

	precision	recall	f1-score	support
car	0.96	0.96	0.96	47184
pedestrian	0.96	0.95	0.95	8448
bus	0.94	0.92	0.93	1435
truck	0.97	0.97	0.97	52366
motorcycle	0.84	0.76	0.80	711
barrier	0.98	0.99	0.98	3631
bicycle_rack	1.00	0.99	1.00	591
accuracy			0.96	114366
macro avg	0.95	0.93	0.94	114366
weighted avg	0.96	0.96	0.96	114366

Şekil 36. $K = 3$ için sınıf bazlı doğruluk raporu

$k=11$ koşulunda modelin genel doğruluğu %95,96'ya düşmüş, özellikle otobüs ve motosiklet sınıflarında duyarlılık değerlerinin azaldığı gözlenmektedir. Bu durum, fazla komşunun karara katılmasıyla nadir sınıfların baskın sınıflara doğru yanlış etiketlenme eğilimini artırdığını göstermektedir.

Veri boyutu: (571830, 3) (571830,)

KNN eğitimi (fit) başlıyor...

Doğruluk: 0.9596

Sınıf Bazlı Rapor:

	precision	recall	f1-score	support
car	0.96	0.96	0.96	47184
pedestrian	0.95	0.92	0.94	8448
bus	0.93	0.85	0.89	1435
truck	0.97	0.97	0.97	52366
motorcycle	0.89	0.67	0.76	711
barrier	0.97	0.99	0.98	3631
bicycle_rack	0.99	1.00	1.00	591
accuracy			0.96	114366
macro avg	0.95	0.91	0.93	114366
weighted avg	0.96	0.96	0.96	114366

Şekil 37. K = 11 için sınıf bazlı doğruluk raporu

Veri boyutu: (571830, 3) (571830,)

KNN eğitimi (fit) başlıyor...

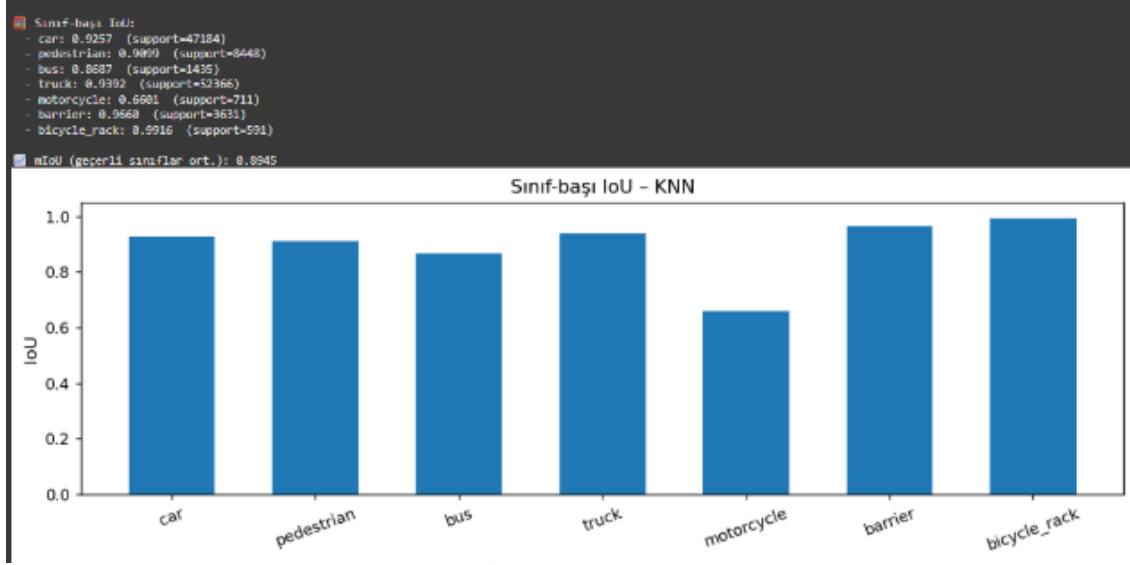
Doğruluk: 0.9616

Sınıf Bazlı Rapor:

	precision	recall	f1-score	support
car	0.96	0.96	0.96	47184
pedestrian	0.95	0.94	0.94	8448
bus	0.94	0.89	0.91	1435
truck	0.97	0.97	0.97	52366
motorcycle	0.86	0.69	0.77	711
barrier	0.97	0.99	0.98	3631
bicycle_rack	1.00	1.00	1.00	591
accuracy			0.96	114366
macro avg	0.95	0.92	0.93	114366
weighted avg	0.96	0.96	0.96	114366

Şekil 38. K = 7 için sınıf bazlı doğruluk raporu

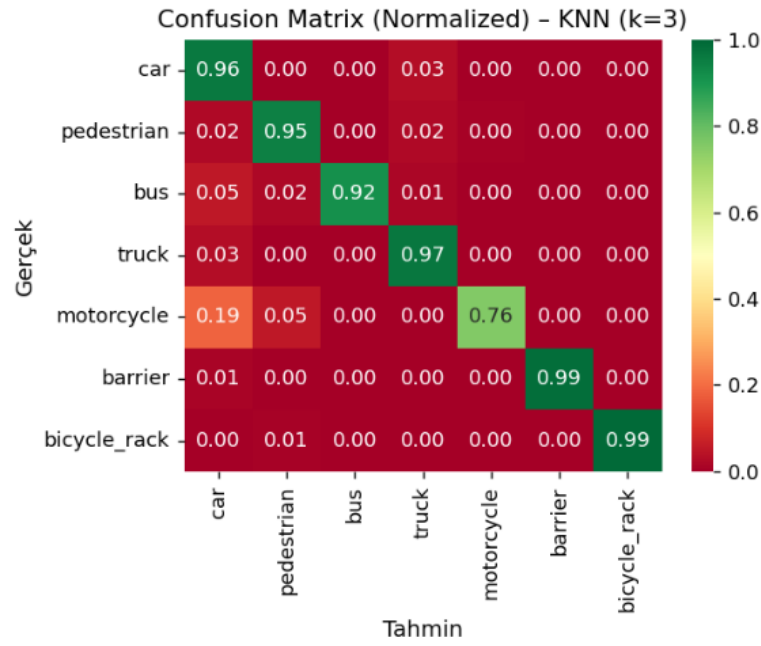
k=7 koşulunda doğruluk oranı %96,16 olarak bulunmuş ve bu değer k=11'e göre daha yüksek performans sağlamıştır. Ancak yine de az örneğe sahip sınıflarda k=3'e kıyasla daha düşük duyarlılık değerleri elde edilmiştir. KNN deneylerinde model performansı sadece doğruluk üzerinden değil, aynı zamanda IoU metriği ile de değerlendirilmiştir. IoU, tahmin edilen ve gerçek etiket kümelerinin kesişim–birleşim oranını ölçen bir metriktir.



Şekil 39. Sınıf bazlı IoU değeri

Çalışmanın bu aşamasında KNN için hesaplanan IoU değerleri sınıflar arasında farklılık göstermiştir. Örneğin araba, kamyon, bariyer ve bisiklet rafı sınıflarında IoU 90% üzeri başarı sağlarken örnek sayısı bu sınıflara göre daha az olan otobüs ve özellikle motosiklet sınıflarında belirgin düşüşler görülmektedir. Bu durum, KNN'in nadir sınıfları sınırlı komşu bilgisiyle ayırt etme güçlüğüne ortaya koymaktadır.

Genel ortalama alındığında KNN'in mIoU değeri yaklaşık olarak %89 olup bu değer sınıf bazlı farklılıklara rağmen modelin genel olarak güçlü bir sınıflandırma yeteneğine sahip olduğunu göstermektedir. Ancak doğruluk metriği olan $k=3$ ile karşılaştırıldığında IoU değerlerinin daha düşük olduğu gözlenmiştir. Bu nedenle, deneylerin geri kalanında IoU yerine doğruluk metriği baz alınmıştır. Ancak sınıf bazlı performans farklılıklarını daha net görebilmek için IoU ek analiz aracı olarak kullanılmıştır. $k=3$ koşulu hem en yüksek doğruluğu sağlamış hem de IoU sonuçlarında daha dengeli bir dağılım göstermiştir. Dolayısıyla KNN modeli, doğruluk metriği baz alınarak $k=3$ parametresi ile çalışmanın ana deneylerine dahil edilmiştir.



Şekil 40. K = 3 için karışıklık matrisi

Şekil 40'ta sunulan karışıklık matrisi, modelin sınıf bazlı başarılarını ayrıntılı biçimde ortaya koymaktadır. Normalize edilmiş matrise göre araba sınıfı %96 oranında doğru tahmin edilmiştir. Yaya sınıfında %95 başarı elde edilmiştir ancak az sayıda örnek araba olarak yanlış sınıflandırılmıştır. Otobüs sınıfı %92 doğrulukla tahmin edilmiştir ancak bazı örneklerin kamyon sınıfı ile karıştırıldığı gözlenmiştir. Kamyon sınıfında %97'lik yüksek doğruluk dikkat gözükmemektedir. Motosiklet sınıfında ise başarı %76'ya düşmüştür bu durum, sınıfın düşük örnek sayısı sebebiyle modelin kararsız kalmasından kaynaklanmaktadır. Bariyer ve bisiklet rafı sınıflarında ise model %99'a varan yüksek doğruluk elde etmiştir. Bu tablo, KNN modelinin özellikle düşük özellikli sınıflarda hata yapma eğiliminde olduğunu, ancak genel olarak güçlü bir sınıf ayrımı sağladığını göstermektedir.

Veri boyutu: (571830, 3) (571830,)

KNN eğitimi (fit) başlıyor...

✓ Doğruluk: 0.9636

Sınıf Bazlı Rapor:

	precision	recall	f1-score	support
car	0.96	0.96	0.96	47184
pedestrian	0.96	0.95	0.95	8448
bus	0.94	0.92	0.93	1435
truck	0.97	0.97	0.97	52366
motorcycle	0.84	0.76	0.80	711
barrier	0.98	0.99	0.98	3631
bicycle_rack	1.00	0.99	1.00	591
accuracy			0.96	114366
macro avg	0.95	0.93	0.94	114366
weighted avg	0.96	0.96	0.96	114366

Şekil 41. K= 3 için sınıf bazlı kesinlik/ duyarlılık/ F1 skor

Şekil 41’de sınıf bazlı kesinlik, duyarlılık ve F1 skorları karşılaştırılmıştır. Araç, kamyon bariyer gibi baskın sınıflarda tüm metriklerde %95 üzeri yüksek performans elde edilmiştir. Yaya sınıfında kesinlik %96, duyarlılık %95, F1 ise %95 olarak ölçülmüş ve dengeli bir sonuç vermiştir. Otobüs sınıfında kesinlik %94, duyarlılık %92 olup, duyarlılık değerinin daha düşük kalması bu sınıfın daha çok yanlış ürettiğini göstermektedir. Motosiklet sınıfında kesinlik %84 olmasına rağmen duyarlılık %76’da kalmıştır bu da modelin motosikletleri çoğunlukla diğer sınıflarla karıştırdığını ortaya koymuştur.

K=3 parametresiyle çalışan KNN modeli, %96.36 genel doğruluk ve büyük sınıflarda oldukça yüksek başarı göstermiştir. En zayıf performans, örnek sayısının azlığı nedeniyle motosiklet sınıfında görülmüştür. Karışıklık matrisleri ve sınıf bazlı metrikler incelendiğinde modelin genelleme yeteneğinin güçlü olduğu ve sınıflar arasında dengeli bir performans sunduğu görülmektedir

2.8.5. PointNet – Mini Yöntemi ile Deneysel Sonuçlar

Bu çalışmada kullanılan yöntemlerden biri PointNet Mini modelidir. Bu model ile 7 sınıf için test kümesinde 25 epoch ile %84.95 doğruluk elde edilmiştir.

🔴 Test Doğruluk: 0.8495

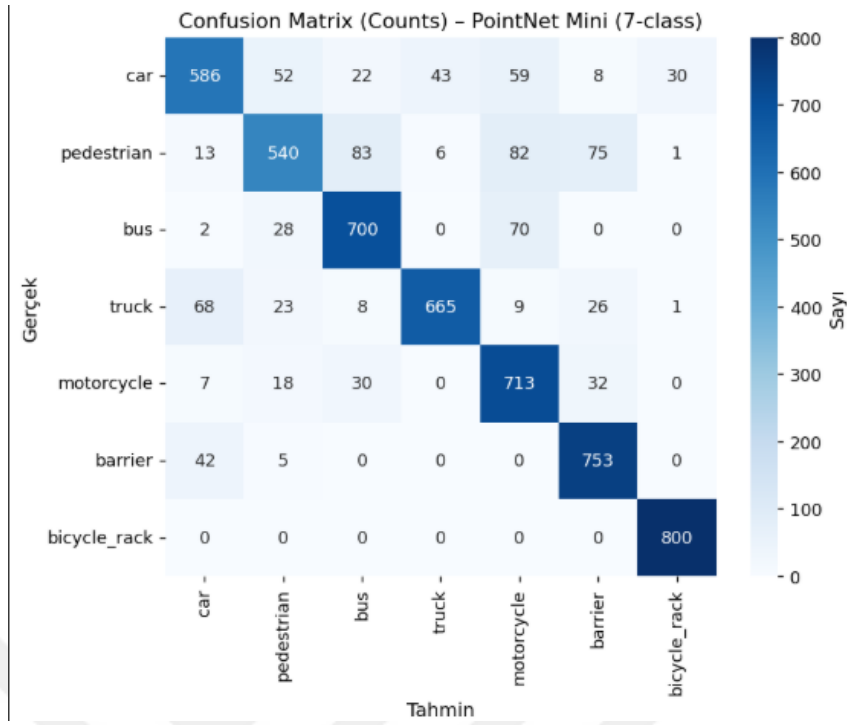
📊 Sınıf Bazlı Rapor:

	precision	recall	f1-score
car	0.8162	0.7325	0.7721
pedestrian	0.8108	0.6750	0.7367
bus	0.8304	0.8750	0.8521
truck	0.9314	0.8313	0.8785
motorcycle	0.7642	0.8912	0.8229
barrier	0.8423	0.9413	0.8890
bicycle_rack	0.9615	1.0000	0.9804
accuracy			0.8495
macro avg	0.8510	0.8495	0.8474
weighted avg	0.8510	0.8495	0.8474

Şekil 42. PointNet-Mini sınıf bazlı doğruluk raporu

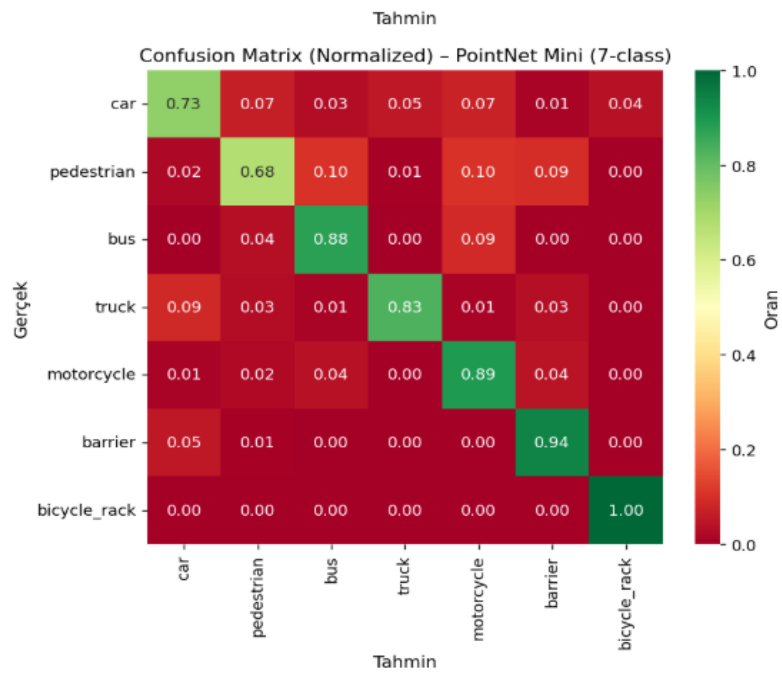
Sınıflara göre performans sonucu incelendiğinde araba sınıfında modelin F1 skoru %77 düzeyinde olduğu gözükmemektedir. Kesinlik yüksek olmasına karşın duyarlılık değerinin %73'e gerilemesi, otomobillerin bir kısmının diğer sınıflarla karıştırıldığını göstermektedir. Yaya sınıfında da araba ile benzer bir durum gözükmemektedir. Küçük nesne boyutu ve Lidar nokta yoğunluğunun düşük olması nedeniyle model yaya örneklerinin bir bölümünü kaçırmıştır. Otobüs sınıfında duyarlılık %87,5 seviyesindedir. Bu durum modelin otobüsleri kaçırmadığını, ancak kesinlik %83 seviyesinde kaldığı için bazı yanlış tahminler ürettiğini göstermektedir. Kamyon sınıfında F1 skoru %88 olup kesinlik ve duyarlılık değerleri dengeli kalmıştır. Model, kamyonları otobüslere göre daha doğru ayırt edebilmiştir. Motosiklet sınıfı diğerlerine nazaran daha düşük başarıya sahiptir. Burada kesinlik %76'da kaldığı gözükmemekte yani bazı örnekler hatalı biçimde motosiklet olarak sınıflandırılmıştır. Bariyer sınıfı F1 = %89 ile yüksek bir başarıya sahiptir. Bu değer duyarlılık değeri olan %94 ile modelin bariyerleri kaçırmadan doğru etiketleyebildiğini göstermektedir. Bisiklet rafı sınıfında ise %98 F1 ile yüksek doğruluk başarısı elde edilmiştir.

Sonuçlara bakıldığında, az örnekli sınıflarda kesinlik düşüşü, fazla örnekli sınıflarda ise duyarlılık düşüşü görülmektedir. Bu durum veri dengesizliği ile ilişkilidir.



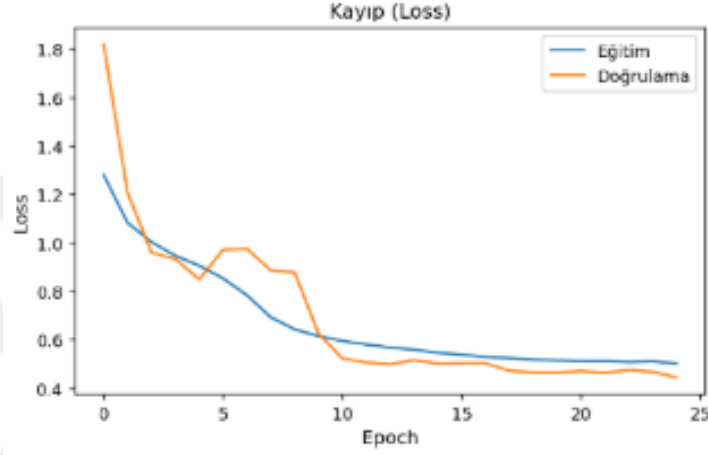
Şekil 43. PointNet-Mini 7 sınıf için karışıklık matrisi

Sayı temelli matriste sınıflar arası karışmaları görselleştirilmiştir. Burada özellikle araba ve yaya sınıflarının birbirine karıştığı, otobüs sınıfının ise bazı örneklerinin kamyon olarak etiketlendiği görülmektedir.



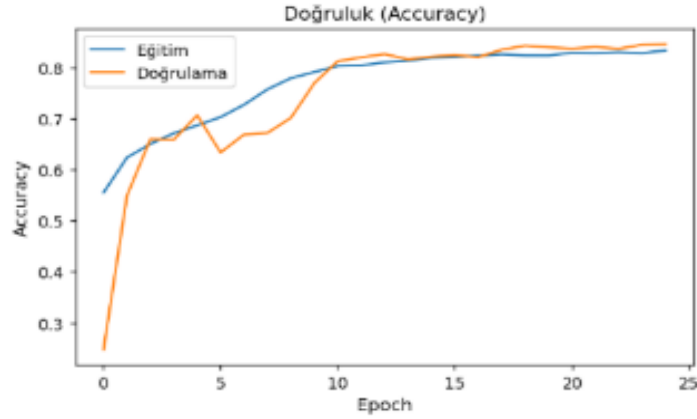
Şekil 44. 7 sınıf için normalize edilmiş karışıklık matrisi

Normalize edilmiş karışıklık matrisi incelendiğinde bu durum daha net gözükmemektedir. Örneğin, yaya sınıfı %68 doğrulukla tanımlanırken, %10 otobüs ve %9 bariyer sınıfına hatalı kaymıştır. Benzer olarak, araba sınıfı %73 doğrulukla tanımlanmış, ancak %7 yaya ve %7 bariyer ile karışmıştır. Bu kod çıktısında küçük nesnelerin ve benzer geometrik özelliklere sahip sınıflarda modelin ayırım yapmakta zorlandığını gösterilmektedir.



Şekil 45. PointNet-Mini Kayıp Grafiği

Eğitim süreci incelendiğinde eğitim ve doğrulama kayıplarının başlangıçta yüksek seviyelerde başladığı, özellikle ilk 10 epoch boyunca hızlı bir düşüş gösterdiği görülmektedir. Bu durum, modelin veri içerisindeki döngüleri kısa sürede öğrenmeye başladığını ortaya koymaktadır. Epoch sayısı arttıkça kayıp değerlerinin yavaşladığı ve yaklaşık 20. epoch sonrasında 0.4–0.5 bandında sabitlendiği görülmektedir. Eğitim ve doğrulama kaybı eğrilerinin birbirine yakın seyretmesi, modelin aşırı uyum yapmadığını kanıtlamaktadır.



Şekil 46. PointNet Mini doğruluk grafiği

Doğruluk grafiği de benzer şekilde düzenli bir öğrenme süreci sunmaktadır. Başlangıçta %55–65 seviyelerinde olan doğruluk, ilk 15 epoch içerisinde hızlı bir artış göstererek %80–85 seviyelerine kadar çıkmıştır. Bu noktadan sonra doğruluk eğrisi yatay bir seyir izlemiş ve yaklaşık 25 epoch sonunda doğrulama doğruluğu %85 seviyesinde sabitlenmiştir. Eğitim ve doğrulama doğruluklarının birbirine yakın olması, modelin sadece eğitim verisine ezber yapmadığını, aynı zamanda doğrulama verisinde de tutarlı bir başarı gösterdiğini ortaya koymaktadır.

Sonuç olarak, kayıp ve doğruluk grafiklerinden elde edilen bulgular, PointNet Mini'nin nuScenes Lidarseg-mini veri seti üzerinde istikrarlı bir öğrenme gerçekleştirdiğini %85 civarında doğrulama doğruluğuna ulaşarak güvenilir bir performans sergilediğini göstermektedir.

2.8.6. CNN Yöntemi ile Deneysel Sonuçlar

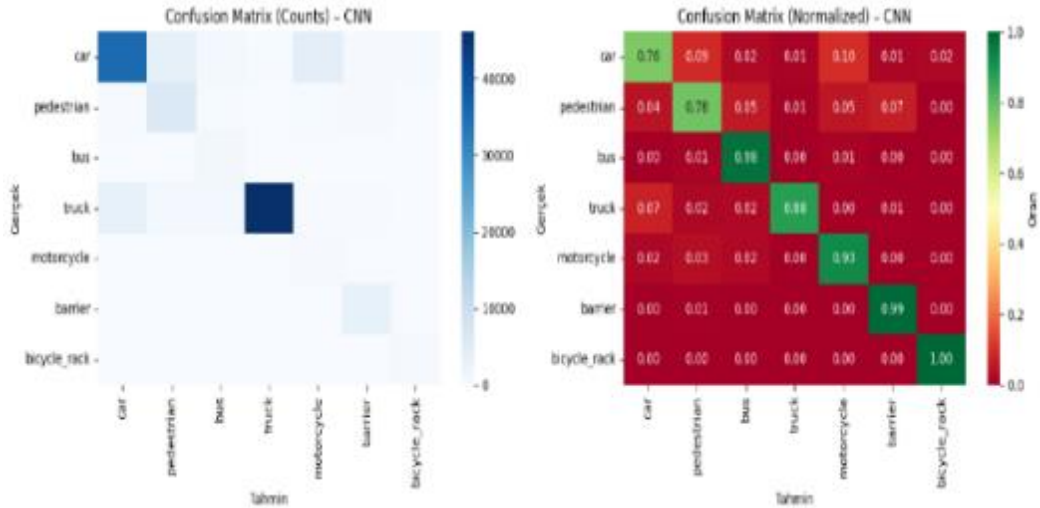
Çalışmada kullanılan modelde, nokta bulutlarının düzenli grid yapısına dönüştürülmesi yerine doğrudan nokta koordinatları kullanılmış ve veriler 1B CNN mimarisi ile işlenmiştir. Eğitim sürecinde Batch Normalization, Dropout ve class weighting teknikleri uygulanarak hem aşırı uyum riski azaltılmış hem de az örnekli sınıfların daha dengeli öğrenmesi sağlanmıştır. Model, tüm veri seti üzerinde 40 epoch boyunca eğitilmiş ve sonuçlar hem doğruluk hem de IoU metrikleri üzerinden değerlendirilmiştir. Özellikle araba ve kamyon gibi veri yoğun sınıflarda yüksek başarı elde edilirken, az örnekli sınıflarda performansın daha düşük kaldığı gözlemlenmiştir.

Sınıf Bazlı Rapor:

	precision	recall	f1-score	support
car	0.90	0.76	0.82	47184
pedestrian	0.56	0.78	0.65	8448
bus	0.36	0.98	0.52	1436
truck	0.99	0.88	0.93	52366
motorcycle	0.11	0.93	0.20	711
barrier	0.71	0.99	0.83	3631
bicycle_rack	0.43	1.00	0.60	591
accuracy			0.83	114367
macro avg	0.58	0.90	0.65	114367
weighted avg	0.89	0.83	0.85	114367

Şekil 47. CNN sınıf bazlı başarı raporu

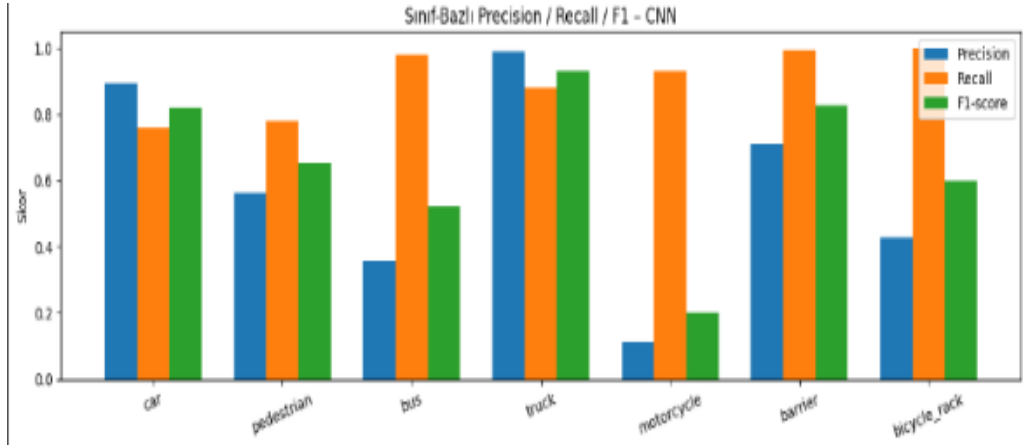
Şekil 47’de CNN modeliyle yapılan deneylerde elde edilen sınıf bazlı performans metrikleri sunulmuştur. Genel doğruluk değeri %83 seviyesinde gerçekleşmiştir. Özellikle kamyon ve araba sınıfları gibi verisi yoğun sınıflarda model yüksek başarı sağlamıştır. Buna karşın, motosiklet ve bisiklet rafı sınıfları gibi az örnek içeren sınıflarda performansın belirli bir şekilde düştüğü görülmektedir. Bu durum, veri dengesizliğinin CNN üzerinde etkili olduğunu göstermektedir.



Şekil 48. CNN 7 sınıf için karışıklık matrisi

CNN modeline ait sayı bazlı karışıklık matrisi incelendiğinde her sınıf için doğru ve yanlış tahmin edilen örneklerin mutlak sayısını verilmektedir. Burada özellikle araba ve kamyon sınıflarında diğerler sınıflara göre daha doğru tahminler elde edilmiştir. Ancak bu sınıfların az da olsa yaya veya diğer sınıflarla karıştığı da görülmektedir.

Görsele ek normalize edilmiş karışıklık matrisi incelendiğinde tahmin performansını oran bazında göstermektedir. Normalize edilmiş matriste en belirgin fark araba ve kamyon sınıflarında yüksek doğruluk oranlarının elde edilmiş olmasıdır. Ancak motosiklet ve bisiklet rafı sınıflarında modelin zorluk yaşadığı görülmektedir. Motosiklet sınıfında %93 oranında doğru tahmin yapılmasına rağmen, düşük kesinlik değerleri bu sınıfa farklı sınıflardan hatalı örneklerin atandığını göstermektedir. Benzer şekilde, bisiklet rafı sınıfı sayıca küçük olduğundan modelin bu sınıfı öğrenmesi güçleşmiş, ancak tespit edilen örneklerin neredeyse tamamı doğru sınıflandırılmıştır. Bariyer sınıfı ise normalize edilmiş matriste %99'a yakın doğruluk oranı ile öne çıkmaktadır. Bu durum, bariyerlerin geometrik yapılarının diğer sınıflardan kolayca ayrılabilmesini ve modelin bu sınıfı yüksek güvenle tanıyabildiğini göstermektedir. Genel olarak karışıklık matrisi sonuçları, CNN modelinin veri yoğun sınıflarda yüksek başarı sağladığını ancak nadir sınıflarda sınıf dengesizliği nedeniyle sınırlı performans gösterdiğini ortaya koymaktadır. Bu durum, eğitim sürecinde kullanılan class weight uygulamasının etkili olsa da örnek sayısı çok düşük sınıflarda nedeniyle sınırlı performans gösterdiğini ortaya koymaktadır. Bu durum, eğitim sürecinde kullanılan class weight uygulamasının etkili olsa da örnek sayısı çok düşük sınıflarda tamamen yeterli olmadığını göstermektedir.

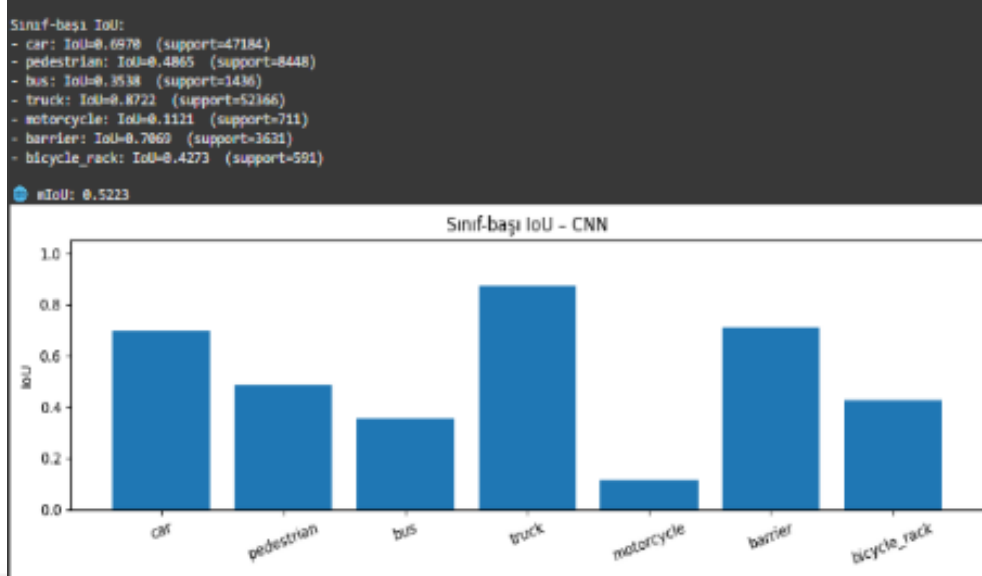


Şekil 49. Cnn her sınıf için skor değerleri

Şekil 49'da görülen çubuk grafikte CNN modelinin her sınıf için kesinlik, duyarlılık ve F1-skoru değerleri gösterilmektedir. Bu kod çıktısı ile modelin sınıflar bazında güçlü ve zayıf yönleri net bir şekilde gözlemlenebilmektedir. Örneğin araba sınıfında kesinlik değeri %90 ile oldukça yüksekken, duyarlılık değeri ise %76 seviyesinde kalmıştır. Bu durum, modelin otomobil sınıfına ait tahminlerde doğruya çok yakın olduğunu ancak bazı gerçek

otomobilleri diğer sınıflar ile karıştırdığını göstermektedir. Yaya sınıfında duyarlılık %78 ile güçlüdür, fakat kesinlik %56 seviyesinde kalmıştır. Bu sonuç modelin çoğu yaya örneğini doğru tespit ettiğini ancak yanlış pozitiflerin sayısının da yüksek olduğunu ortaya koymaktadır. Otobüs sınıfı, duyarlılık açısından %98 ile en yüksek başarıya sahiptir. Ancak kesinlik değeri %36'ya kadar düşmüştür. Yani otobüslerin neredeyse tamamı yakalanmış olsa da çok sayıda farklı sınıf hatalı olarak otobüs kategorisine atanmıştır. Kamyon sınıfı, hem kesinlik %99 oranı hem de duyarlılık %88 oranı ile dengeli ve yüksek başarı göstermiştir. Bu sonuç, kamyonların geometrik olarak diğer sınıflardan daha net ayrılabilmesini göstermektedir. Motosiklet sınıfı, duyarlılık açısından %93 gibi oldukça yüksek bir başarıya sahip olmasına rağmen kesinlik değeri %11'de kalmıştır. Bu sınıfta model birçok gerçek motosikleti doğru tahmin etse de farklı sınıflardan hatalı atamalar yapılmıştır. Bariyer sınıfı, kesinlik %71 ve duyarlılık %99 değerleri ile güçlü performans sergilemiştir. Özellikle duyarlılık değerinin yüksekliği, bariyer örneklerinin model tarafından gözden kaçırılmadığını ortaya koymaktadır. Bisiklet rafı sınıfında duyarlılık %100 seviyesindedir, ancak kesinlik %43'te kalmıştır. Bu durum, modelin tüm bisiklet park yerlerini doğru yakaladığını ancak farklı sınıflardan gelen bazı noktaları da bu kategoriye yanlış atadığını göstermektedir.

Genel olarak incelendiğinde, grafik modelin baskın sınıflarda daha dengeli ve yüksek performans sergilediğini ancak nadir sınıflarda ise hatalı sınıf atamalarının arttığını göstermektedir. Bu tablo, veri dengesizliğinin CNN performansını doğrudan etkilediğini ve özellikle kesinlik değerlerinde ciddi farklılıklara yol açtığını ortaya koymaktadır.



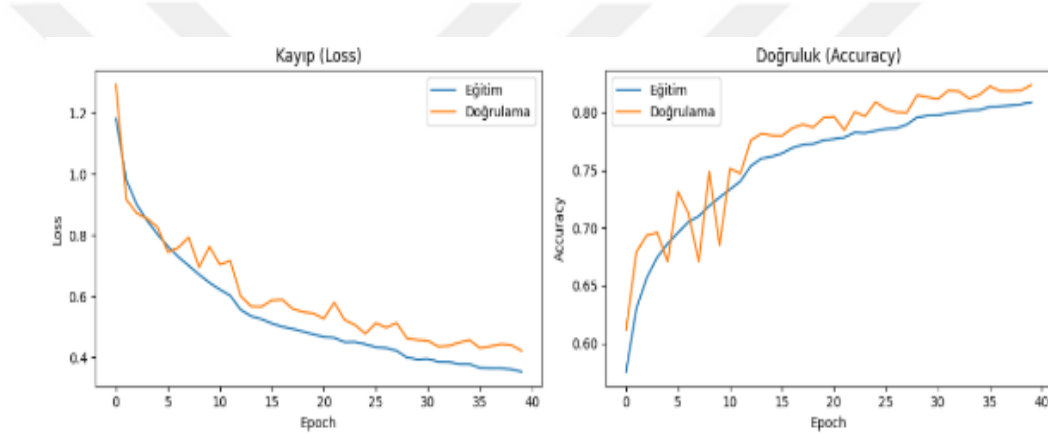
Şekil 50. Sınıf bazlı IoU değeri

Şekil 50’de CNN modelinin sınıf bazında Intersection over Union (IoU) skorlarını göstermektedir. IoU metriği, nesne tanıma ve segmentasyon problemlerinde sıkça kullanılan, tahmin edilen alan ile gerçek alanın kesişim ve birleşim oranını ölçen bir değerlendirme kriteridir. Bu çalışmada, IoU değerleri her sınıf için ayrı ayrı hesaplanmış ve ayrıca ortalama IoU değeri de elde edilmiştir.

Sonuçlar incelendiğinde kamyon sınıfı, %87,22 gibi oldukça yüksek bir IoU değerine ulaşarak en başarılı sınıf olmuştur. Bu durum, kamyonların geometrik yapılarının diğer sınıflardan daha belirgin olmasıyla açıklanabilir. Araba sınıfı da %69,70 IoU ile güçlü bir performans sergilemiştir. Bu değer, modelin araçları yüksek doğrulukla ayırt edebildiğini göstermektedir. Bariyer sınıfında elde edilen %70,69’luk IoU değeri, bariyerlerin yapısal özelliklerinin model tarafından kolayca öğrenilebildiğini ortaya koymaktadır. Yaya ve otobüs sınıflarında ise performans %48,65 ve %35,38 ile diğerlerine göre daha düşüktür. Bunun temel nedeni, bu sınıfların nokta bulutu içindeki temsil oranlarının daha düşük olması ve geometrik çeşitlilikten kaynaklanan zorluklardır. Motosiklet sınıfı, yalnızca %11,21 IoU ile en zayıf sonucu vermiştir. Bu değer, motosikletlerin nokta bulutunda düşük yoğunluklu ve küçük yapılara sahip olması sebebiyle model tarafından ayırt edilmesinin zorlaştığını göstermektedir. Bisiklet rafı sınıfında ise %42,73 IoU elde edilmiştir. Bu sonuç orta seviyede bir başarıya işaret etmekte, ancak sınırlı örnek sayısı nedeniyle hatalı atamaların etkili olduğunu göstermektedir. Ortalama IoU değeri %52,23 olarak hesaplanmıştır. Bu değer, CNN modelinin genel olarak orta seviyede bir alan örtüşme

başarısı sağladığını göstermektedir. Ancak bazı sınıflarda yüksek IoU değerleri elde edilirken, özellikle küçük ve nadir sınıflarda performans düşüşleri görülmüştür. Bu CNN'in küçük sınıfları ayırt etmede zorlandığını ortaya koymaktadır. Bu sorunu hafifletmek amacıyla çalışmada class weight yaklaşımı kullanılmıştır. Yani, her sınıfın eğitim sırasında modele katkısı, veri setindeki örnek sayısı ile ters orantılı olacak şekilde ayarlanmıştır. Bu yöntem, nadir sınıfların daha fazla önemszenmesini sağlamış ve modelin bu sınıflar için kısmen daha yüksek başarıya ulaşmasına yardımcı olmuştur.

Buna rağmen, CNN'in özellikle küçük nesneler üzerinde IoU değerlerinin düşük kalması, bu tür sınıflar için ileri seviye veri artırma yöntemleri veya nokta tabanlı derin öğrenme mimarilerinin daha uygun olabileceğini göstermektedir.



Şekil 51. Cnn modeli eğitim ve doğrulama grafiği

Şekil 51'de CNN modelinin eğitim ve doğrulama için elde edilen kayıp ve doğruluk eğri grafikleri görülmektedir. Kayıp grafiği incelendiğinde başlangıçta hem eğitim hem doğrulama kaybının 1.2'nin üzerinde başladığı ve ilk birkaç epoch içerisinde hızlı bir şekilde düşüş gösterdiği görülmektedir. 10. epoch'tan sonra kayıplar daha yavaş bir şekilde azalmaya devam etmiş ve yaklaşık 0.3–0.4 seviyelerinde dengelenmiş olduğu görülmüştür. Grafik incelendiğinde eğitim ve doğrulama kaybı arasında belirgin bir kopma görülmemektedir. Bu durum, modelin aşırı öğrenme yapmadığını, genel olarak kararlı bir öğrenme süreci izlediğini göstermektedir. Doğruluk grafiği incelendiğinde eğitim doğruluğunun ilk epoch'ta %60 seviyesindeyken, 10. epoch civarında %75'e ulaştığı görülmektedir. Doğrulama doğruluğu ise benzer bir eğilim göstermiş ve bazı dalgalanmalara rağmen eğitim doğruluğuna yakın seyretmiştir. Bu dalgalanmalar, doğrulama setinin bazı sınıflarındaki örneklerin azlığı ve dengesizliğiyle açıklanabilir. Son

epochlarda eğitim doğruluğu %81, doğrulama doğruluğu ise %83 seviyesine ulaşmıştır. Eğitim ve doğrulama doğruluklarının birbirine yakın olması, modelin genelleme kabiliyetinin yüksek olduğunu göstermektedir.

CNN modeli, yaklaşık 40 epoch boyunca istikrarlı bir şekilde öğrenme göstermiştir. Hem kayıp hem doğruluk eğrilerinde gözlenen paralellik, modelin aşırı öğrenme veya yetersiz uyum sorunu yaşamadığını ve hiperparametre seçimlerinin uygun olduğunu kanıtlamaktadır. Doğrulama setindeki küçük dalgalanmalar küçük sınıflarda modelin zorlandığını gösterse de genel performansı etkileyecek düzeyde değildir.

2.9. Ek Denemeler ve Deneylere Dahil Edilmeyen Çalışmalar

Bu bölümde çalışma kapsamında yapılmış ancak asıl deneylere eklenmemiş olan çalışmalara yer verilmiştir.

2.9.1. Lidar Nokta Bulutu ve Kamera Görüntüleri Eşleştirme Denemeleri

Bu çalışmada, ek analiz olarak Lidar nokta bulutu verileri ile kamera görüntüleri eşleştirilmeye çalışılmıştır. Burada amaç sınıflandırma sonucunda elde edilen 3B noktaların kamera düzleminde görsel olarak doğrulanması olmuştur. NuScenes veri setinde her kamera için kalibrasyon parametreleri sağlandığından, teorik olarak her Lidar noktası kamera koordinat sistemine dönüştürülebilmekte ve ilgili görüntü pikseline yansıtılabilmektedir. Denemede her bir Lidar noktası öncelikle sensörler arası dönüşüm matrisleri kullanılarak kamera koordinat sistemine çevrilmiştir ardından kamera iç parametreleri yardımıyla 2B görüntü düzlemine aktarılmıştır. Böylece sınıflandırılmış Lidar noktalarının, gerçek görüntüde hangi piksellere karşılık geldiği elde edilmiştir. Denemeler yaya sınıfı öncelikli olmak üzere farklı nesne sınıfları için yapılmıştır.

□ LiDAR → Görüntü Projeksiyonu (pedestrian)



Şekil 52. Yaya sınıfına ait noktaların ön kamera görüntüleri ile eşleşmesi

Şekil 52’de yaya sınıfına ait Lidar noktalarının ön kamera görüntüsü üzerine yansımaları gösterilmektedir. Beklenen durum, yaya konumunda kümelenmiş noktaların görüntüdeki yaya bölgesi ile örtüşmesidir. Ancak noktaların büyük çoğunluğu yol üzerinde veya yanlış konumlarda görselleşmiştir. Bu durumun temel sebebi, Lidar ve kamera sensörlerinin araç üzerinde farklı yükseklik ve açılarda konumlanmış olmasıdır. Araçlarda kameralar, Lidar sensörüne göre daha yukarıda yer almakta, bu da eşleşme sırasında perspektif sapmalarına yol açmaktadır.

CAM_BACK_LEFT — Nokta: (1076, 480)



Şekil 53. Arka sol kamera görüntüsünde yaya noktasının eşleşmesi

Benzer şekilde Şekil 53'te yaya sınıfına ait bir noktanın arka sol kamera görüntüsünde projeksiyonu görülmektedir. Noktanın yaya üzerinde konumlandığı gözükse de sapma dikkat çekmektedir.

CAM_BACK_LEFT — Nokta: (1077, 480)



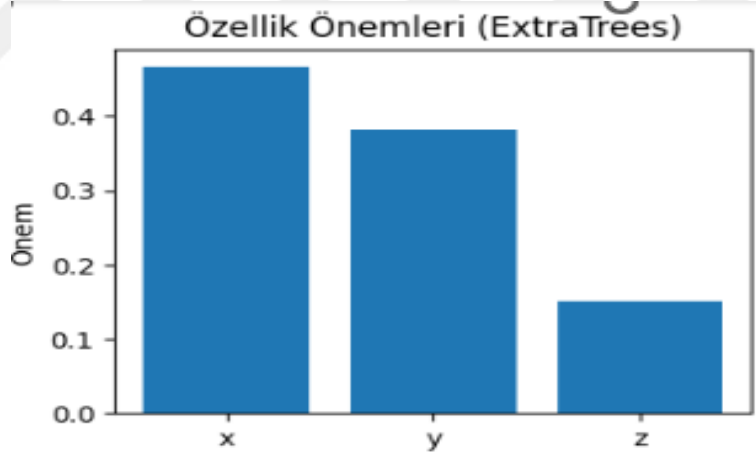
Şekil 54. Karanlık ortamda Lidar noktasının kamera görüntüsüne eşleşmesi

Şekil 54 ise gece koşullarında elde edilmiş bir örneği göstermektedir. Burada düşük ışık ve kamera gürültüsü nedeniyle Lidar noktalarının projeksiyonu görsel olarak ayırt edilemez hale gelmiştir. Kamera görüntüsündeki kalite düşüklüğü Lidar verisinin

doğrulanmasını daha da zorlaştırmıştır. Bu denemenin deneylere koyulmamasının sebepleri arasında kamera ve Lidar'ın farklı yüksekliklerde ve açılarda monte edilmesi ile nokta bulutu ve görüntü arasında tam hizalanmama sorununun olması, veri setinde zaman damgaları eşleşse bile hızlı hareket eden nesnelerde projeksiyon hatalarının büyümesi, gece sahneleri veya parlak ışık kontrastlarının kamera verisinin kalitesini düşürmesi ve eşleşmenin doğrulanmasını zorlaştırması yer almaktadır. Bu nedenlerle, Lidar-kamera eşleştirme çalışmaları deneylerin ana gövdesine dahil edilmemiştir. Elde edilen çıktılar, sensör füzyonunun önemini göstermekte, ancak kullanılan basit projeksiyon yaklaşımı NuScenes-mini alt kümesinde yeterli başarıyı sağlayamamıştır.

2.9.2. Extra Tree (Karar Ağaçları) Yöntemi ile Deneme

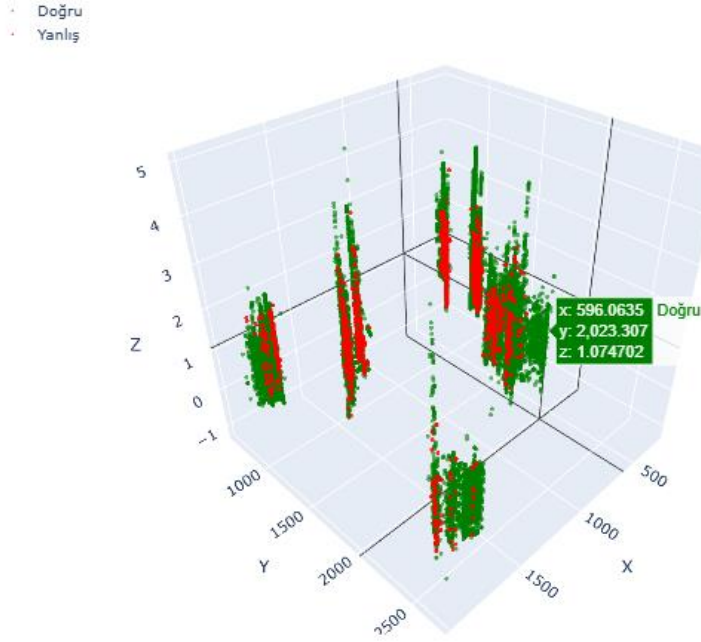
Bu çalışmada ayrıca, Rastgele Orman (Random Forest, RF) algoritması ile aynı aileden gelen Extra Trees (Extremely Randomized Trees) yöntemi de denenmiştir. Extra Trees, RF'den farklı olarak ağaç yapısını oluştururken daha fazla rastgelelik içermektedir.



Şekil 55. Özellik önemleri grafiği

Şekil 55'te görülen özellik önemleri grafiği, modelin nokta bulutu özniteliklerinden özellikle x ve y koordinatlarını ayırt edici bulunduğunu göstermektedir. X eksenini yaklaşık %45, y eksenini %38 önem taşıırken, dikey konum bilgisini temsil eden z eksenini yalnızca %15 katkı sağlamıştır. Bu durum, nesne sınıflandırmada yatay düzlemdeki konumun daha baskın bir rol oynadığını göstermektedir.

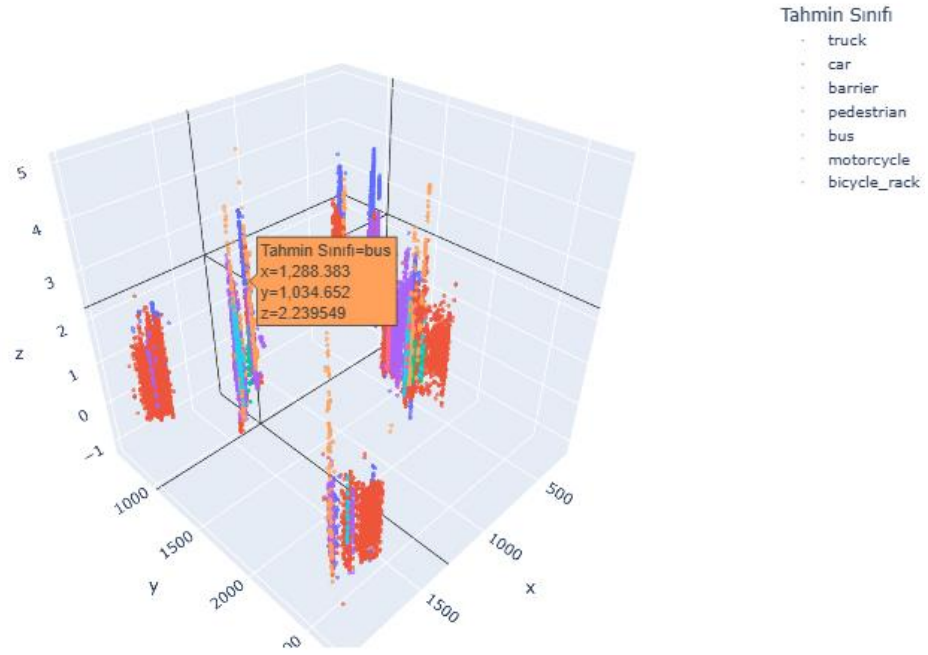
ExtraTrees – Doğru (yeşil) / Yanlış (kırmızı) Sınıflandırılan Noktalar (3B)



Şekil 56. Doğru ve yanlış sınıflandırılan noktalar

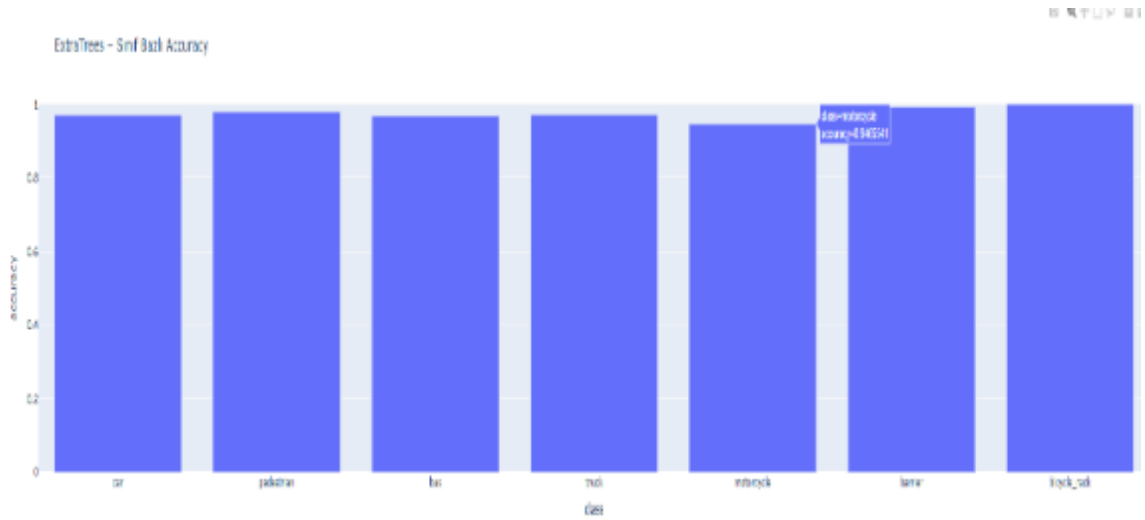
Şekil 56’da yer alan 3B görselleştirmede doğru sınıflandırılan noktalar yeşil, yanlış sınıflandırmalar ise kırmızı renk ile gösterilmiştir. Bu kod çıktısında modelin genel olarak yoğun bölgelerde başarı sağladığı ancak kenar gibi zor bölgelerde ve küçük nesnelerde çok sayıda yanlış tahmin yaptığı görülmektedir. Özellikle otobüs ve motosiklet sınıflarında yanlış etiketlemeler belirgin olarak gözükmemektedir.

ExtraTrees – Tahmin Edilen Sınıfa Göre 3B Nokta Bulutu



Şekil 57. Tahmin edilen sınıflara göre 3B nokta bulutları

Şekil 57’de modelin tahmin edilen sınıflara göre 3B nokta bulutu gösterilmektedir. Farklı renkler farklı sınıfları temsil etmekte olup, görselleştirme otobüs ve kamyon gibi sınıfların çakıştığını gözükmemektedir. Bu da modelin benzer geometrik yapıya sahip nesneleri ayırt etmekte zorlandığını göstermektedir.



Şekil 58. Sınıf bazlı doğruluk grafiği

Şekil 58’de sınıf bazlı doğruluk oranları incelendiğinde, araba, yaya ve kamyon sınıflarında yüksek doğruluk elde edildiği, ancak motosiklet sınıfında başarının %95’in altına düştüğü görülmektedir. Bu durum küçük örnekli sınıflar için modelin kararsızlığını ortaya koymaktadır.

Extra Trees sınıflandırma yöntemi diğer yöntemlere göre hızlı eğitim süresi ve öznitelik önemlerinin kolay yorumlanabilirliği açısından avantaj sağlamıştır. Ancak modelin performansı Random Forest ile karşılaştırıldığında doğruluk oranlarında üstün bir özellik göstermemiştir. Bu nedenle, Extra Trees algoritması Random Forest ile aynı karar ağaçları ailesinden geldiği için ve tekrarlayıcı sonuçlar sunduğundan dolayı asıl deneysel karşılaştırmalara dahil edilmeyerek ek denemeler başlığı altında raporlanmıştır



3. BULGULAR VE TARTIŞMA

3.1. Deney Düzeni

Bu çalışmada yapılan deneylerin temel amacı, Lidar nokta bulutu verilerinin sınıflandırılması için farklı makine öğrenmesi ve derin öğrenme tabanlı yöntemlerin performansını karşılaştırmalı olarak değerlendirme olmuştur. Deneyler boyunca aynı veri kümesi, aynı sınıf sayısı, aynı sınıflar ve benzer giriş özellikleri kullanılarak yöntemler arasında adaletli bir kıyaslama yapılmasına özen gösterilmiştir. Bu yaklaşım sayesinde her algoritmanın güçlü ve zayıf yönleri, doğrudan aynı koşullar altında gözlemlenebilmiştir.

Deneylerde kullanılan veri kümesi, nuScenes lidarseg-mini alt kümesinden elde edilmiştir. Veri, toplamda araba, yaya, otobüs, kamyon, motosiklet, bariyer ve bisiklet rafı olmak üzere 7 farklı sınıfa ait noktalardan oluşmaktadır. Bu sınıflar, otonom sürüş uygulamaları açısından öneme sahip olup, farklı boyutlarda ve farklı yoğunluklarda dağılıma sahiptir. Örneğin, araba ve kamyon gibi sınıflar yüksek sayıda örneğe sahipken, motosiklet ve bisiklet rafı sınıfları diğer sınıflara göre daha az sayıda örnekle temsil edilmiştir. Bu durum, sınıf dengesizliği sorununu ortaya çıkarmış ve deneylerin tasarımında dikkate alınması gereken önemli bir unsur haline gelmiştir.

Verilerde her nokta yalnızca x , y , z koordinat bilgisiyle temsil edilmiştir. Yani yoğunluk veya diğer sensör özellikleri kullanılmamış, tüm yöntemler için aynı girdi yapısı tercih edilmiştir. Bunun nedeni, yöntemlerin özellikle ham geometrik bilgiyi nasıl işlediğini görmek ve farklı algoritmalar arasında doğrudan karşılaştırma yapabilmektir. Böylece ek özniteliklerin etkisi değil, yöntemlerin kendine özgü öğrenme kapasiteleri değerlendirilmiştir.

Veri ön işleme aşamasında nokta bulutları eğitim, doğrulama ve test setleri olarak üçe ayrılmıştır. Bu ayrım stratified sampling yöntemi ile yapılmıştır. Stratified sampling, her bir sınıfın oransal dağılımının hem eğitim hem de test setinde korunmasını sağlamış ve özellikle motosiklet ve bisiklet rafı gibi dengesiz sınıflarda bu yöntem kullanılmadığında, az örnekli sınıfların test setinde hiç yer almaması gibi sorunlar yaşanabilmiştir. Bu nedenle stratified sampling, sonuçların güvenilirliği ve genellenebilirliği açısından önemli bir rol oynamıştır.

Eğitim sürecinde bazı yöntemlerde dengesizliği dengelemek için ek stratejiler uygulanmıştır. Örneğin, MLP modelinde doğrudan sınıf ağırlıklandırması desteklenmediği için aşırı örnekleme kullanılmıştır. Bu sayede az sayıda örneğe sahip sınıflar çoğaltılarak modelin bu sınıfları öğrenme kapasitesi artırılmıştır. Ek olarak, denemelerde CNN modelinde dengesizlik sorunu `class_weight` parametresi ile çözülmeye çalışılmıştır. Böylece nadir sınıflara daha yüksek bir ağırlık verilmiş bu sayede modelin bu sınıfları göz ardı etmesi engellenmiştir.

Deneylerde kullanılan yöntemler hem makine öğrenmesi hem de derin öğrenme yöntemlerinden seçilmiştir. Makine öğrenmesi tarafında KNN, Random Forest ve sklearn MLP değerlendirilirken derin öğrenme tarafında ise PointNet Mini, 1B CNN ve Keras tabanlı MLP test edilmiştir. Bu seçim sayesinde geleneksel yöntemlerle modern derin öğrenme mimarilerinin aynı veri üzerinde ne ölçüde başarılı oldukları incelenebilmiştir. Ayrıca, MLP hem sklearn uygulaması hem de derin öğrenme versiyonu ile çalıştırılmıştır ve böylece aynı yöntem ailesinin farklı uygulamalarının sonuçları da kıyaslanmıştır.

Her yöntemin eğitim parametreleri, veri setinin büyüklüğü ve donanım kısıtları göz önüne alınarak ayarlanmıştır. Örneğin, KNN’de farklı k değerleri (3, 7, 11) denenmiş ve en iyi sonuç $k=3$ ile elde edilmiştir. CNN modelinde eğitim 40 epoch ile sınırlandırılmış, erken durdurma stratejisi kullanılarak aşırı uyum engellenmiştir. PointNet Mini ise hesaplama maliyetini düşürmek amacıyla 25 epoch boyunca eğitilmiştir.

Sonuç olarak, deney düzeni tasarlanırken her yöntem için hem Colab ortamında uygulama kolaylığı hem de adaletli karşılaştırma kriterleri dikkate alınmıştır. Tüm modeller yalnızca x , y , z koordinatlarını giriş olarak almış, sınıf dengesizliği farklı stratejilerle giderilmeye çalışılmış ve eğitim parametreleri Colab ortamının sınırlarına göre optimize edilmiştir. Bu sayede elde edilen sonuçların hem yöntemler arasında hem de literatürdeki benzer çalışmalarla kıyaslanabilir nitelikte olması sağlanmıştır.

3.2. Genel Performans Karşılaştırması

Bu çalışmada uygulanan farklı makine öğrenmesi ve derin öğrenme yöntemleri, aynı veri kümesi üzerinde çalıştırılarak performansları karşılaştırılmıştır. Çalışmada yöntemlerin değerlendirilmesinde öncelikli olarak sınıf-bazlı kesinlik, duyarlılık ve F1-skorları ile karışıklık matrisleri kullanılmıştır. Nokta bulutu sınıflandırması çalışmasında görüntü tabanlı segmentasyondan farklı olarak piksel veya voxel bazlı alan örtüşmesini

değil, tek tek noktaların doğru ya da yanlış sınıflandırılması sağlanmıştır. Bu nedenle, hangi sınıflarda başarı sağlandığını ve hangi sınıflarda karışıklık yaşandığını görmek için en uygun göstergeler bu metrikler olmuştur. Özellikle macro-F1 değeri, örneğin çok sayıda araba kamyon noktasına karşılık, daha az sayıda motosiklet veya bisiklet rafı noktası bulunan yani sınıf dengesizliği olan kümelerde daha adil bir değerlendirme sağlamıştır. Nokta bulutu sınıflandırmasında asıl hedef, tekil noktaların doğru etiketlenmesi olduğu için IoU bu çalışmada diğer metrikler gibi zorunlu bir kıyaslama ölçütü olarak ele alınmamıştır. Bu yüzden sadece destekleyici bilgi amacıyla kullanılmış ve bazı deneylerde sınıf-bazlı IoU sonuçları da raporlanmıştır.

Deneyler sonucunda, yöntemler arasında hem genel doğruluk hem de sınıf bazlı performans metriklerinde farklılıklar ortaya çıktığı görülmüştür. Bu farklılıkların temelinde kullanılan algoritmaların öğrenme becerileri, veri dengesizlikleriyle başa çıkma kapasiteleri ve modelin karmaşıklığı gibi faktörler yer almaktadır.

Tablo 4. Uygulanan yöntemlerin genel doğruluk performansı

Algoritma	Türü	Doğruluk (%)	F1-Skoru (%)
Random Forest (RF)	Makine Öğrenmesi	97,21	98
K-Nearest Neighbors (KNN, k=3)	Makine Öğrenmesi	96,36	94
ML-MLP	Makine Öğrenmesi	90,34	81
DL-MLP	Derin Öğrenme	70,17	45
PointNet Mini	Derin Öğrenme	84,95	84.7
1B CNN	Derin Öğrenme	83	65

Tablo 4'te uygulanan yöntemlerin genel başarı sonuçları incelendiğinde, Random Forest yönteminin %97,21 doğruluk ve %98 F1-skoru ile tüm algoritmalar arasında en yüksek başarıyı elde ettiği görülmektedir. Bu durum, RF algoritmasının karar ağaçları sayesinde karmaşık karar sınırlarını öğrenmede oldukça güçlü bir yöntem olmasından kaynaklanmaktadır.

KNN algoritması da %96,36 doğruluk ve %94 F1-skoru ile oldukça yüksek bir performans sağlamıştır. KNN'nin bu doğruluk başarısı, sınıflar arasındaki mesafeleri doğrudan kullanarak etkili bir karar mekanizması kurabilmesine dayanmaktadır.

Makine öğrenmesi tabanlı MLP yönteminde ise %90,34 doğruluk ve %81 F1-skoru ile diğer yöntemlere nazaran daha mütevazı bir performans görülmektedir. Bunun temel nedeni kullanılan MLP mimarisinin diğerlerine nispeten daha basit olması ve karmaşık geometrik ilişkileri tam olarak yakalayamamasından kaynaklanmaktadır.

Derin öğrenme yöntemlerine bakıldığında, derin MLP modeli %70,17 doğruluk ve sadece %45 F1-skoru ile en düşük performansı sergilemiştir. Bu durum, modelin veri dengesizliğine karşı daha hassas olduğunu ve yeterince karmaşık özellikleri yakalayamadığını göstermektedir.

Buna karşın, nokta bulutu için özel olarak geliştirilen PointNet Mini modeli %84,95 doğruluk ve %84,7 F1-skoru ile derin MLP'ye göre daha başarılı sonuçlar üretmiştir. Özellikle nokta tabanlı girişlerle çalışması, uzamsal bilgiyi daha iyi değerlendirmesine imkân tanımıştır. Ancak, kullanılan mini sürümün parametre sayısı düşük olduğu için tam anlamıyla PointNet performansına ulaşamamıştır.

Son olarak, 1B CNN modeli %83 doğruluk ve %65 F1-skoru ile orta düzeyde bir performans sergilemiştir. CNN'in özellik çıkarma yeteneği sayesinde kamyon, araba gibi belirli sınıflarda yüksek doğruluk sağlansa da motosiklet, bisiklet rafı gibi dengesiz sınıflarda başarısı daha sınırlı kalmıştır.

Bu çalışmada F1-skoru olarak macro ortalama tercih edilmiştir. Bunun nedeni, veri kümesinde ciddi sınıf dengesizliklerinin bulunmasıdır. Örneğin, araba ve kamyon sınıflarında on binlerce nokta bulunurken, motosiklet ve bisiklet rafı gibi sınıflarda nokta sayısı çok daha azdır. Eğer sadece ağırlıklı F1 kullanılsaydı, yüksek örnek sayısına sahip sınıflar ağırlığı bastıracak ve az örnekli sınıflardaki başarısızlıklar göz ardı edilecekti. Macro F1 ise her sınıfa eşit ağırlık vererek, az örnekli sınıfların da performansını yansıtmaktadır. Bu sayede, algoritmaların yalnızca genel başarısı değil, tüm sınıflar üzerindeki dengeli performansı da değerlendirilebilmiştir.

3.2.1. Araba Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması

Çalışmanın bu bölümünde her bir sınıf için farklı algoritmaların elde ettiği başarılar ayrı ayrı incelenmiştir. Burada amaç, yalnızca genel doğruluk değerleri üzerinden değil, aynı zamanda her sınıfın tahmin performansını kesinlik, duyarlılık ve F1 skoru bağlamında karşılaştırmaktır. Böylece hangi algoritmanın hangi sınıfta ne kadar güçlü ya da ne kadar zayıf kaldığı yorumlanabilmektedir. Öncelikle araba sınıfı için elde edilen sonuçlar Tablo 5'te sunulmuş ve çıktılar karşılaştırmalı olarak değerlendirilmiştir.

Tablo 5. Araba sınıfı performans sonuçları

Sınıf	Algoritma	Tür	Kesinlik (%)	Duyarlılık (%)	F1 Skoru (%)
Araba	Random Forest	Makine Öğrenmesi	97	97	97
Araba	KNN	Makine Öğrenmesi	96	96	96
Araba	ML-MLP	Makine Öğrenmesi	93	87	90
Araba	DL-MLP	Derin Öğrenme	87,6	56	68
Araba	CNN	Derin Öğrenme	90	76	82
Araba	PointNet Mini	Derin Öğrenme	81,6	73,2	77,2

Tablo 5 incelendiğinde Random Forest modeli, %97 kesinlik, %97 duyarlılık ve %97 F1 skoru ile araba sınıfında en yüksek başarıyı göstermiştir. Bu sonuç, RF algoritmasının veri dağılımı içinde yoğun ve baskın olan araba sınıfını iyi derecede öğrendiğini göstermektedir. KNN algoritması %96 kesinlik, %96 duyarlılık ve %96 F1 skoru ile RF'ye çok yakın performans göstererek ikinci sırada yer almıştır. Bu, KNN'nin özellikle büyük sınıflarda güçlü bir ayırıcı olduğunu ortaya koymuştur. Makine öğrenmesi tabanlı MLP, %93 kesinlik ve %90 F1 skoru ile iyi denilebilecek sonuçlar verirken duyarlılık değerinin %87'de kalması bazı araba örneklerini kaçırdığını göstermektedir. Derin öğrenme tabanlı

MLP, araba sınıfında belirgin şekilde düşük performans sergilemektedir. Duyarlılık değerinin %56'ya kadar düşmesi, modelin araba sınıfını sıklıkla yanlış sınıflarla karıştırdığını göstermektedir. Bu da F1 skorunu %68'e kadar düşürmüştür. CNN modeli, %90 kesinlik ve %82 F1 skoru ile derin öğrenme yöntemleri arasında daha dengeli bir performans sunmaktadır. Özellikle duyarlılık değerinin %76 olması, modelin araba örneklerinin çoğunu doğru tespit ettiğini ancak bir kısmını yanlış sınıflandırdığını göstermektedir. PointNet Mini, %81,6 kesinlik ve %77,2 F1 skoru ile diğer yöntemlere göre daha düşük başarı elde etmiştir. Bunun nedeni, kullanılan mini versiyonun sınırlı parametre kapasitesidir. Ancak duyarlılık değerinin %73,2 olması, modelin araba örneklerini büyük oranda doğru tespit ettiğini fakat kesinlik açısından zayıf kaldığını göstermektedir.

Sonuç olarak, araba sınıfı için özellikle RF ve KNN makine öğrenmesi yöntemleri en yüksek performansı sergilemiştir. Derin öğrenme yöntemleri ise araba sınıfında diğerlerine görece daha düşük başarı göstermiştir. Bu durum, özellikle veri setinde yoğun örneklerle sahip olan araba sınıfının, ağaç tabanlı ve mesafe tabanlı makine öğrenmesi yöntemleri tarafından daha kolay öğrenildiğini ortaya koymaktadır.

3.2.2. Yaya Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması

Çalışmanın bu bölümünde yaya sınıfı için algoritmaların elde ettiği başarılar ayrı ayrı incelenmiştir. Elde edilen sonuçlar Tablo 6'da sunulmuş ve çıktılar karşılaştırmalı olarak değerlendirilmiştir.

Tablo 6. Yaya sınıfı performans sonuçları

Sınıf	Algoritma	Tür	Kesinlik (%)	Duyarlılık (%)	F1 Skoru (%)
Yaya	Random Forest	Makine Öğrenmesi	98	98	98
Yaya	KNN	Makine Öğrenmesi	96	95	95
Yaya	ML-MLP	Makine Öğrenmesi	73	94	82
Yaya	DL-MLP	Derin Öğrenme	37,7	47	42
Yaya	CNN	Derin Öğrenme	56	78	65
Yaya	PointNet Mini	Derin Öğrenme	81	67.5	73.6

Random Forest, yaya sınıfında %98 kesinlik, %98 duyarlılık ve %98 F1 skoru ile en yüksek başarıyı göstermiştir. Bu sonuç, RF'nin yaya sınıfını hem doğru tespit ettiğini hem de yanlış sınıflandırmadan kaçınmakta güçlü olduğunu ortaya koymuştur. Veri dengesizliğine rağmen yayaları yüksek oran ile doğru şekilde ayırabilmesi, modelin karar ağaçları üzerinden sınıf sınırlarını iyi öğrenmesinden kaynaklanmaktadır. KNN algoritması, %96 kesinlik, %95 duyarlılık ve %95 F1 skoruyla oldukça yüksek bir başarı ortaya koymuştur. RF'ye göre biraz daha düşük olsa da KNN'nin mesafe tabanlı yapısı yayaları küçük karışıklıklarla doğru sınıflandırabilmiştir. Makine öğrenmesi tabanlı MLP, %73 kesinlik, %94 duyarlılık ve %82 F1 skoru ile orta seviyede bir başarı göstermiştir. Kesinliğin görece düşük olması, modelin bazı diğer sınıfları yanlışlıkla yaya olarak etiketlediğini gösterirken yüksek %94 duyarlılık değeri ise çoğu yayanın doğru tahmin edilebildiğini göstermektedir. Derin öğrenme tabanlı MLP %37,7 ile bu sınıfta oldukça düşük performans sergilemiştir. Kesinliğin %37,7, duyarlılığın %47 ve F1 skorunun %42'de kalması, modelin yayaları ayırt etmede zorlandığını ve sınıflar arası ciddi karışıklık yaşadığını göstermektedir. Özellikle duyarlılığın düşük olması, yayaların büyük bir kısmının yanlış sınıflarla karıştırıldığını göstermektedir. CNN modeli, %56 kesinlik, %78 duyarlılık ve %65 F1 skoru ile derin öğrenme tabanlı yöntemler arasında nispeten daha başarılı olmuştur. Duyarlılık değerinin %78 çıkması, modelin yayaların çoğunu doğru

şekilde bulunduğunu gösterirken kesinliğin düşük olması nedeniyle başka sınıfları yanlışlıkla yaya olarak tahmin ettiği yorumlanabilmektedir. PointNet Mini, %81 kesinlik, %67,5 duyarlılık ve %73,6 F1 skoruyla CNN'den daha farklı bir sonuç sergilemiştir. Kesinlik değerinin daha yüksek olması, modelin tahmin ettiği yayaların çoğunun gerçekten de yaya sınıfında olduğunu gösterirken duyarlılığın %67,5'te kalması bazı yayaların gözden kaçırıldığını ortaya koymaktadır. Bu nedenle PointNet Mini, CNN'e göre daha güvenilir ancak daha tespitleri daha eksik yapan bir model olarak yorumlanabilir.

Yaya sınıfı, araba sınıfı gibi daha baskın sınıflara kıyasla daha zor bir sınıflandırma sunmuştur. Özellikle RF, KNN gibi makine öğrenmesi tabanlı algoritmalar bu sınıfta da en yüksek performansı göstermiş ve neredeyse kusursuz sınıflandırma sağlamıştır. Ancak derin öğrenme yöntemleri daha düşük performanslar göstermişlerdir. CNN daha çok yayayı yakalamış ama yanlış pozitif üretmiştir. PointNet Mini ise yayalarda daha seçici davranmıştır ancak bazı yayaları atlamıştır. DL-MLP ise en zayıf sonuçları ortaya koymuştur. Bu tablo, derin öğrenme yöntemlerinin daha fazla veri ve parametreye ihtiyaç duyduğunu gösterirken makine öğrenmesi yöntemlerinin ise bu veride daha doğru ve güvenilir sonuçlar verdiğini ortaya koymuştur.

3.2.3. Otobüs Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması

Çalışmanın bu bölümünde otobüs sınıfı için algoritmaların elde ettiği başarılar ayrı ayrı incelenmiştir. Elde edilen sonuçlar Tablo 7'de sunulmuş ve çıktılar karşılaştırmalı olarak değerlendirilmiştir.

Tablo 7. Otobüs sınıfı performans sonuçları

Sınıf	Algoritma	Tür	Kesinlik (%)	Duyarlılık (%)	F1 Skoru
Otobüs	Random Forest	Makine Öğrenmesi	97	96	96
Otobüs	KNN	Makine Öğrenmesi	94	92	93
Otobüs	ML-MLP	Makine Öğrenmesi	57	99	73
Otobüs	DL-MLP	Derin Öğrenme	17	67.6	27
Otobüs	CNN	Derin Öğrenme	36	98	52
Otobüs	PointNet Mini	Derin Öğrenme	83	87.5	85

Tablo 7 incelendiğinde Random Forest yöntemi otobüs sınıfında da en yüksek performansı gösteren algoritma olmuştur. %97 kesinlik, %96 duyarlılık ve %96 F1 skoruyla model hem otobüsleri doğru tespit etmiş hem de diğer sınıfları karıştırmadan güvenilir tahminler yapabilmektedir. KNN algoritması %94 kesinlik, %92 duyarlılık ve %93 F1 skoru ile RF'ye yakın bir başarı sağlamıştır. Makine öğrenmesi tabanlı MLP, bu sınıfta %57 kesinlik gibi düşük bir değere karşın, %99 duyarlılık elde edilmiştir. Bu sonuç modelin neredeyse tüm otobüsleri doğru yakaladığını, ancak diğer sınıflardan çok sayıda örneği yanlışlıkla otobüs olarak sınıflandırdığını göstermektedir. Bu dengesizlik nedeniyle F1 skoru %73'te kalmıştır. Derin öğrenme tabanlı MLP yöntemi otobüs sınıfında en zayıf sonuçları üretmiştir. Kesinlik yalnızca %17 duyarlılık ise %67,6 seviyesinde kalmıştır. Bu sonuçlar modelin otobüsleri genellikle bulamadığını, bulduğu durumlarda da yanlış pozitif oranının çok yüksek olduğunu ortaya koymaktadır. F1 skorunun %27 gibi düşük bir değere inmesi, modelin bu sınıfta başarısız olduğunu net bir şekilde göstermektedir. CNN modeli, %36 kesinlik ve %98 duyarlılık değerleriyle her otobüsü bulmaya çalışmış ancak bunu yaparken çok fazla yanlış pozitif üreten bir yaklaşım sergilemiştir. Duyarlılığın çok yüksek olması, otobüslerin gözden kaçmadığını gösterirken kesinliğin düşük olması, yanlış tahmin

oranının yüksek olduğunu göstermektedir. Bu nedenle F1 skoru %52'ye kadar düşmüştür. PointNet Mini, %83 kesinlik, %87,5 duyarlılık ve %85 F1 skoruyla CNN ve DL-MLP'ye göre oldukça başarılı olmuştur. Model, otobüsleri hem yeterli düzeyde tespit etmiş hem de yanlış sınıflandırmaları sınırlı düzeyde tutabilmiştir. Özellikle CNN'e kıyasla daha dengeli kesinlik-duyarlılık değerleri sergilemesi görülmektedir.

Otobüs sınıfı, veri miktarı daha düşük olan bir kategori olduğu için algoritmalar arasında farklı performanslar gözlenmiştir. Makine öğrenmesi yöntemleri otobüs sınıfında da derin öğrenme yöntemlerine göre üstünlüklerini korumuştur. Yüksek kesinlik ve duyarlılık değerleriyle neredeyse kusursuz sınıflandırma sağlamıştır. Genel olarak, RF ve KNN otobüs sınıfında en başarılı yöntemler olurken, PointNet Mini derin öğrenme tarafında öne çıkmıştır. CNN ve DL-MLP ise daha zayıf kalmıştır.

3.2.4. Kamyon Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması

Çalışmanın bu bölümünde kamyon sınıfı için algoritmaların elde ettiği başarılar ayrı ayrı incelenmiştir. Elde edilen sonuçlar Tablo 8'de sunulmuş ve çıktılar karşılaştırmalı olarak değerlendirilmiştir.

Tablo 8. Kamyon sınıfı performans sonuçları

Sınıf	Algoritma	Tür	Kesinlik (%)	Duyarlılık (%)	F1 Skoru
Kamyon	Random Forest	Makine Öğrenmesi	97	97	97
Kamyon	KNN	Makine Öğrenmesi	97	97	97
Kamyon	ML-MLP	Makine Öğrenmesi	99	92	95
Kamyon	DL-MLP	Derin Öğrenme	95	84	90
Kamyon	CNN	Derin Öğrenme	99	88	93
Kamyon	PointNet Mini	Derin Öğrenme	93	83	87.9

Random Forest ve KNN yöntemleri kamyon sınıfında da neredeyse kusursuz bir performans sergilemiştir. Her iki algoritma da %97 kesinlik, %97 duyarlılık ve %97 F1 skoru ile kamyonları yüksek doğrulukla sınıflandırmıştır. Bu sonuç, makine öğrenmesi yöntemlerinin güçlü genelleme kapasitesini ve veri dengesizliğine karşı dayanıklılığını doğrulamaktadır. ML tabanlı MLP %99 doğruluğuyla kesinlik açısından yüksek bir başarı göstermiştir. Ancak duyarlılık değeri %92’de kalmıştır. Bu sonuç modelin yanlış pozitif oranını çok düşürürken bazı kamyonları atladığını göstermektedir. F1 skoru %95 ile Random Forest ve KNN’den daha düşük kalsa da oldukça yüksek bir performans sergilemiştir. Derin öğrenme tabanlı MLP %95 kesinlik değeriyle iyi denilebilecek bir sonuç elde etse de duyarlılığın %84’e düşmesi, modelin kamyon sınıfındaki tüm örnekleri tespitte zorlandığını göstermektedir. Bu nedenle F1 skoru %90 seviyesinde kalmıştır. CNN modeli, 99% kesinlik oranı ile kamyonları doğru sınıflandırma konusunda çok başarılı olmuştur. Ancak duyarlılık değeri %88’e düşmüştür. Yani model yanlış pozitifleri büyük ölçüde azaltırken bazı kamyon örneklerini kaçırmıştır. Sonuç olarak F1 skoru %93 olarak gerçekleşmiştir. PointNet Mini, %93 kesinlik, %83 duyarlılık ve %87,9 F1 skoruyla diğer yöntemlere göre daha düşük performans göstermiştir. Özellikle duyarlılık değerinin daha düşük kalması, modelin kamyon örneklerini tam olarak kapsamakta zorlandığını göstermektedir.

Kamyon sınıfı sonuçları incelendiğinde özellikle Random Forest ve KNN’in performansı yine ön plana çıkmaktadır. Bu iki yöntem, kamyon sınıfında dengeli kesinlik-duyarlılık değerleri sayesinde en yüksek performansa ulaşmıştır. ML-MLP de yüksek kesinlik avantajıyla güçlü bir alternatif olurken duyarlılık değerinin biraz düşük kalması genel başarıyı sınırlamıştır.

Derin öğrenme tabanlı yöntemlerde ise CNN güçlü bir kesinlik sergilemiştir ancak duyarlılık değerinde kayıplar yaşamıştır. DL-MLP ve PointNet Mini ise makine öğrenmesi yöntemlerine kıyasla daha düşük performans göstermiştir.

Sonuç olarak, kamyon sınıfında makine öğrenmesi yöntemleri açık şekilde üstün performans sağlamış, derin öğrenme tabanlı yaklaşımlar ise daha düşük düzeyde kalmıştır.

3.2.5. Motosiklet Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması

Çalışmanın bu bölümünde motosiklet sınıfı için algoritmaların elde ettiği başarılar ayrı ayrı incelenmiştir. Elde edilen sonuçlar Tablo 9’da sunulmuş ve çıktılar karşılaştırmalı olarak değerlendirilmiştir.

Tablo 9. Motosiklet sınıfı performans sonuçları

Sınıf	Algoritma	Tür	Kesinlik (%)	Duyarlılık (%)	F1 Skoru
Motosiklet	Random Forest	Makine Öğrenmesi	97	93	95
Motosiklet	KNN	Makine Öğrenmesi	84	76	80
Motosiklet	ML-MLP	Makine Öğrenmesi	16	96	28
Motosiklet	DL-MLP	Derin Öğrenme	5.7	74	10.6
Motosiklet	CNN	Derin Öğrenme	11	93	20
Motosiklet	PointNet Mini	Derin Öğrenme	76.4	89	82

Tablo incelendiğinde Random Forest yönteminin motosiklet sınıfında en yüksek başarıyı gösterdiği görülmektedir. %97 kesinlik ve %93 duyarlılık değerleri ile dengeli bir performans ile %95 F1 skoru elde etmiştir. Bu sonuç, algoritmanın motosiklet örneklerini hem doğru hem de istikrarlı şekilde sınıflandırabildiğini göstermiştir. KNN yöntemi %84 kesinlik ve %76 duyarlılık ile kısmen orta düzeyde bir başarı sağlamıştır. Kesinlik değerinin duyarlılık değerine göre daha yüksek olması, modelin yanlış pozitifleri azaltabilmesine rağmen birçok motosikleti atladığını göstermektedir. F1 skoru %80 ile Random Forest’ın belirgin şekilde gerisinde kalmıştır. ML tabanlı MLP de duyarlılık değeri %96 gibi oldukça yüksek olsa da kesinlik yalnızca %16 seviyesindedir. Bu durum, modelin neredeyse tüm motosikletleri motosiklet olarak etiklediğini ancak çok fazla

yanlış pozitif ürettiğini göstermektedir. Dolayısıyla F1 skoru yalnızca %27'ye düşmüştür. DL-MLP, %5,7 kesinlik değeri ile ciddi bir düşüş yaşamıştır, duyarlılık ise %74 düzeyinde kalmıştır. Bu nedenle F1 skoru yalnızca %11 gibi çok düşük bir seviyeye gerilemiştir. Bu sonuç, derin öğrenme tabanlı MLP'nin motosiklet sınıfında başarısız olduğunu ortaya koymaktadır. CNN, duyarlılık değeri açısından %93 ile oldukça yüksek bir başarı elde etmiştir. Ancak kesinlik %11 seviyesindedir. Bu durum, modelin motosiklet sınıfını tespit eğiliminde olduğunu fakat çok fazla yanlış pozitif ürettiğini göstermektedir. F1 skoru %20 seviyesinde kalmıştır. PointNet Mini, motosiklet sınıfında derin öğrenme yöntemleri arasında en iyi sonucu vermiştir. %76,4 kesinlik ve %89 duyarlılık değerleriyle dengeli bir performans ortaya koyarken %82 F1 skoru ile KNN'in önüne geçmiştir. Ancak Random Forest'a kıyasla geride kalmıştır.

Motosiklet sınıfı sonuçları, sınıfın dengesiz veri dağılımı nedeniyle zorlayıcı olduğu görülmektedir. DL-MLP, CNN gibi derin öğrenme tabanlı yöntemler düşük kesinlik değerleriyle başarısız kalırken, Random Forest bu sınıfta da en iyi sonuçları üretmiştir. PointNet Mini, derin öğrenme tabanlı yöntemler arasında ve %82 F1 skoru iyi bir performans sağlamıştır. Buna karşılık ML-MLP, yüksek duyarlılık avantajına rağmen aşırı düşük kesinlik nedeniyle güvenilir bir sonuç verememiştir. Sonuç olarak motosiklet sınıfında, Random Forest en başarılı yöntem olurken, PointNet Mini de özellikle dengesiz sınıflarda derin öğrenmenin potansiyelini göstermiştir.

3.2.6. Bariyer Sınıfı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması

Çalışmanın bu bölümünde bariyer sınıfı için algoritmaların elde ettiği başarılar ayrı ayrı incelenmiştir. Elde edilen sonuçlar Tablo 10'da sunulmuş ve çıktılar karşılaştırmalı olarak değerlendirilmiştir.

Tablo 10. Bariyer sınıfı performans sonuçları

Sınıf	Algoritma	Tür	Kesinlik (%)	Duyarlılık (%)	F1 Skoru
Bariyer	Random Forest	Makine Öğrenmesi	100	99	99
Bariyer	KNN	Makine Öğrenmesi	98	99	98
Bariyer	ML-MLP	Makine Öğrenmesi	98	99	98
Bariyer	DL-MLP	Derin Öğrenme	42, 5	93	58
Bariyer	CNN	Derin Öğrenme	71	99	83
Bariyer	PointNet Mini	Derin Öğrenme	84	94	89

Tablo 10’ da Random Forest yönteminin bariyer sınıfında en yüksek performansı gösterdiği görülmektedir. Kesinlik %100, duyarlılık %99 ve F1 skoru %99 seviyesindedir. Bu sonuç modelin bariyer sınıfını neredeyse hatasız bir şekilde ayırt edebildiğini göstermektedir. KNN yöntemi %98 kesinlik, %99 duyarlılık ve %98 F1 skoru ile Random Forest’a çok yakın bir performans sergilemiştir. Sadece ufak farklarla geri kalmasına rağmen, bariyer sınıfı için oldukça güvenilir bir model olduğu görülmektedir. ML tabanlı MLP, diğer yöntemlere benzer şekilde %98 kesinlik ve %99 duyarlılık değerleriyle yüksek bir başarıya ulaşmıştır ve %98 F1 skoru elde etmiştir. Bu sonuç bariyer sınıfının makine öğrenmesi yöntemleri için kolay ayrıştırılabilen bir kategori olduğunu ortaya koymaktadır. DL-MLP, bariyer sınıfında oldukça zayıf bir performans sergilemiştir. Duyarlılık değeri %93 ile kötü bir sonuç olmamasına rağmen, kesinlik yalnızca %42,5 seviyesinde kalmıştır. Bu dengesizlik nedeniyle F1 skoru %58’e kadar düşmüştür. Bu sonuç, derin MLP’nin bariyer sınıfında yüksek miktarda yanlış pozitif ürettiğini göstermektedir. CNN yöntemi %99 duyarlılık oranıyla oldukça iyi bir sonuç elde etmiştir yani bariyer noktalarının büyük çoğunluğunu doğru etiketleyebilmiştir. Ancak kesinlik %71 seviyesinde kalmış ve bu

durum modelin bazı diğer sınıfları yanlışlıkla bariyer olarak etiketlediğini göstermektedir. F1 skoru %83'tür ve DL-MLP'den çok daha iyi olmasına rağmen makine öğrenmesi yöntemlerinden geridedir. PointNet Mini, %84 kesinlik ve %94 duyarlılık oranı ile dengeli bir performans sunmuş ve %89 F1 skoru elde etmiştir. CNN'den daha yüksek kesinlik oranı sağlamıştır ve duyarlılık açısından biraz geride kalsa da genel performansı daha dengelidir.

Bariyer sınıfı sonuçları, bu sınıfın makine öğrenmesi yöntemleriyle oldukça başarılı şekilde sınıflandırılabilirdiğini göstermektedir. Random Forest, %99 F1 skoru ile en iyi sonucu elde etmiştir. KNN ve ML-MLP de %98 F1 skoru ile oldukça güçlü performans göstermiştir.

Derin öğrenme yöntemleri arasında PointNet Mini, %89 F1 skoru ile öne çıkarken, CNN %83 ile daha geride kalmıştır. DL-MLP ise bu sınıfta belirgin şekilde başarısız olmuş, yüksek yanlış pozitif oranı nedeniyle yalnızca %58 F1 skorunda kalmıştır. Sonuç olarak bariyer sınıfında makine öğrenmesi yöntemleri, derin öğrenme yaklaşımlarına kıyasla çok daha güçlü ve tutarlı bir performans sergilemiştir.

3.2.7. Bisiklet Rafı için Algoritma Bazlı Doğruluk ve F1 Karşılaştırması

Çalışmanın bu bölümünde bisiklet rafı sınıfı için algoritmaların elde ettiği başarılar ayrı ayrı incelenmiştir. Elde edilen sonuçlar Tablo 11'de sunulmuş ve çıktılar karşılaştırmalı olarak değerlendirilmiştir.

Tablo 11. Bisiklet rafı sınıfı performans sonuçları

Sınıf	Algoritma	Tür	Kesinlik (%)	Duyarlılık (%)	F1 Skoru
Bisiklet Rafı	Random Forest	Makine Öğrenmesi	100	100	100
Bisiklet Rafı	KNN	Makine Öğrenmesi	100	99	100
Bisiklet Rafı	ML-MLP	Makine Öğrenmesi	100	100	100
Bisiklet Rafı	DL-MLP	Derin Öğrenme	12	97,6	22
Bisiklet Rafı	CNN	Derin Öğrenme	43	100	60
Bisiklet Rafı	PointNet Mini	Derin Öğrenme	96	100	98

Tablo 11 incelendiğinde Random Forest yöntemi bisiklet rafı sınıfında %100 kesinlik, %100 duyarlılık ve %100 F1 skoru elde ederek mükemmel bir başarı sağlamıştır. Model bu sınıfı hiç hata yapmadan sınıflandırabilmiştir. KNN de %100 kesinlik ve %99 duyarlılık ile yine neredeyse kusursuz bir performans göstermiş, F1 skoru %100'e ulaşmıştır. Sadece küçük bir miktar yanlış negatif üretilmiştir. ML tabanlı MLP, %100 kesinlik ve %100 duyarlılık ile Random Forest'a eş değer bir sonuç elde etmiştir ve F1 skorunda %100 seviyesine ulaşmıştır. Bu sonuç, makine öğrenmesi yöntemlerinin bisiklet rafı sınıfını rahatça ayırt edebildiğini göstermektedir. DL-MLP, bu sınıfta en düşük performansı göstermiştir. Duyarlılık değeri %97,6 gibi yüksek bir seviyede olmasına rağmen kesinlik yalnızca %12'de kalmıştır. Yani model, bisiklet rafı sınıfına ait olmayan birçok noktayı yanlışlıkla bu sınıfa atamıştır. Bu dengesizlik, F1 skorunun %22 gibi oldukça düşük bir seviyeye inmesine neden olmuştur. CNN, %100 duyarlılık değerine ulaşarak tüm bisiklet rafı noktalarını doğru tahmin edebilmişken kesinlik yalnızca %43'te kalmıştır. Bu da çok sayıda yanlış pozitif ürettiğini göstermektedir. Dolayısıyla F1 skoru %60 seviyesine düşmüştür. PointNet Mini, %96 kesinlik ve %100 duyarlılık ile derin

öğrenme yöntemleri arasında en dengeli ve başarılı sonucu vermiştir. F1 skoru %98'e ulaşmış, makine öğrenmesi yöntemlerinin performansına oldukça yakın bir başarı göstermiştir.

Bisiklet rafı sınıfı için en yüksek başarıyı yine makine öğrenmesi yöntemleri elde etmiştir. Random Forest ve ML-MLP, %100 F1 skoru ile hatasız çalışırken, KNN de %100'e çok yakın bir performans göstermiştir. Derin öğrenme yöntemleri arasında ise PointNet Mini, %98 F1 skoru ile öne çıkarken CNN %60 ile orta düzeyde kalmıştır. DL-MLP ise bu sınıfta başarısız olmuştur.

Bu sonuç, bisiklet rafı sınıfının düşük örnek sayısına rağmen belirgin geometrik özelliklere sahip olduğunu, bu nedenle makine öğrenmesi algoritmalarıyla çok iyi ayırt edilebildiğini göstermektedir. Derin öğrenme tabanlı modeller ise özellikle dengesiz sınıflarda yüksek yanlış pozitif üretmeye daha yatkın görünmektedir.

3.3. Genel Sonuçlar ve Karşılaştırma

Bu çalışmada yapılan deneylerin tümü dikkate alındığında, yöntemler arasında hem genel doğruluk hem de sınıf-bazlı başarılar açısından farklılıklar gözlemlenmiştir. Bu bölümde algoritmaların genel performansları özetlenmekte ve yöntemlerin güçlü ve zayıf yönleri sunulmaktadır.

Makine öğrenmesi yöntemleri arasında Random Forest diğerlerine göre açık ara en başarılı sonuçları vermiştir. Yaklaşık %97 doğruluk ve %98 F1-skoru ile diğer yöntemlerin önüne geçmiştir. RF'nin bu başarısı, ağaç tabanlı yapısının düzensiz dağılıma sahip nokta bulutu verisine uyum sağlayabilmesi ve özellikle dengesiz sınıflarda kararlı sonuçlar üretebilmesinden kaynaklanmaktadır. Özellikle araba, kamyon ve bariyer sınıflarında yüksek tutarlılık elde edilmiştir.

KNN, $k = 3$ düşük komşu sayısında %96 doğruluk düzeyine ulaşırken az örnekli sınıflarda duyarlılık değerlerinde kayıplar yaşamıştır. Makine öğrenmesi tabanlı MLP, %90 civarında doğrulukla RF ve KNN'in gerisinde kalmıştır ancak sade yapısına rağmen güçlü performans göstermiştir.

Derin öğrenme yöntemleri arasında PointNet Mini, özellikle bisiklet rafı ve bariyer sınıflarında yüksek F1-skorları üretmiş, ancak yaya ve motosiklet gibi küçük sınıflarda makine öğrenmesi yöntemlerinin gerisinde kalmıştır. Bu durum hem sınıf dengesizliği hem de modele giriş olarak kullanılan nokta sayısının sınırlı tutulmasıyla ilişkilidir. CNN

modeli, yaklaşık %83 doğruluk ile daha düşük genel performans sergilemiş olsa da kamyon ve otobüs sınıflarında geometrik ilişkileri yakalayabilmesi sayesinde kararlı tahminler yapabilmektedir. Derin öğrenme tabanlı MLP ise %70 civarında doğruluk sağlamış, fakat RF ve KNN'in oldukça gerisinde kalmıştır.

Genel değerlendirmede, Random Forest en başarılı yöntem olarak öne çıkmıştır. Bu sonuç, nokta bulutu sınıflandırmasında özellikle dengesiz veri kümelerinde ağaç tabanlı yöntemlerin avantajlı olduğunu göstermektedir. Öte yandan, CNN ve PointNet Mini gibi derin öğrenme yöntemleri daha büyük veri kümeleri, daha derin mimariler veya veri artırma teknikleri ile güçlendirildiğinde potansiyel olarak çok daha yüksek performans sergileyebilir.



4. SONUÇLAR VE ÖNERİLER

Bu tez çalışmasında, nuScenes lidarseg-mini veri seti kullanılarak nokta bulutu sınıflandırması gerçekleştirilmiş ve hem makine öğrenmesi hem de derin öğrenme tabanlı çeşitli yöntemlerin performansları karşılaştırılmıştır. Çalışmanın temel amacı, farklı algoritmaların nokta bulutu verisi üzerindeki sınıflandırma başarısını incelemek ve yöntemler arasında güçlü ve zayıf yönleri ortaya koymaktır.

Uygulanan yöntemler arasında Random Forest, KNN, ML-MLP, DL-MLP CNN ve PointNet Mini yer almıştır. Tüm yöntemler 7 sınıf için değerlendirilmiş ve doğruluk, kesinlik, duyarlılık, F1-skoru ile karışıklık matrisleri üzerinden analiz edilmiştir.

Elde edilen bulgulara göre, Random Forest yaklaşık %97 doğruluk ve %98 F1 ile en başarılı yöntem olmuştur. Bu sonuç, özellikle dengesiz sınıflar üzerinde bile kararlı tahminler yapabilen ağaç tabanlı yöntemlerin bu problemde güçlü bir çözüm sunduğunu göstermektedir. KNN, düşük komşu sayılarında %96 başarı ile Random Forest'e rekabetçi sonuçlar elde etmiş ancak sınıf sayısı arttıkça hassasiyeti düşmüştür. ML-MLP %90 doğruluk ile basit yapısına rağmen güçlü bir alternatif olmuş, fakat RF'nin gerisinde kalmıştır.

Derin öğrenme yöntemlerinden PointNet Mini, ham nokta bulutunu doğrudan işleme kapasitesi sayesinde bisiklet rafı ve bariyer gibi bazı sınıflarda yüksek başarı sağlamış, ancak küçük örnekle sınıflarda zorlanmıştır. CNN, %83 doğrulukla genel olarak orta performans göstermiş, fakat kamyon ve otobüs gibi sınıflarda geometrik ilişkileri yakalayabilmiştir. Derin öğrenme MLP ise %70 doğruluk ile düşük bir başarı sağlayarak makine öğrenmesi yöntemlerinin gerisinde kalmıştır.

Genel olarak, bu çalışmada makine öğrenmesi tabanlı yöntemlerin, özellikle Random Forest'in, mevcut veri kümesinde derin öğrenme tabanlı modellere kıyasla daha yüksek doğruluk sağladığı görülmüştür. Bunun temel nedenleri arasında veri kümesinin görece küçük boyutu, sınıf dengesizliği ve derin öğrenme yöntemlerinin daha büyük veri ve daha uzun eğitim süreçlerine ihtiyaç duyması bulunmaktadır.

5. KAYNAKLAR

- Alaba, O. A., & Ball, J. E. (2022). Deep learning approaches for 3D LiDAR point cloud object classification: A survey. *Sensors*, 22(3), 1121.
- Atik, M., Güngör, Ö., & Keskin, E. (2022). Fotogrametrik nokta bulutu verisinin makine öğrenmesi ile sınıflandırılması. *Journal of Geodesy and GeoInformation*, 9(2), 137–149.
- Azizmalayeri, F., Peyghambarzadeh, S. M. M., Khotanlou, H., & Salarpour, A. (2018). Kernel correlation based CNN for point cloud classification task. In *2018 8th International Conference on Computer and Knowledge Engineering (ICCCKE)* (pp. 200–204). IEEE.
- Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.
- Bisong, E. (2019). *Building machine learning and deep learning models on Google Cloud Platform: Apress, Berkeley, CA*.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32.
- Buda, M., Maki, A., & Mazurowski, M. A. (2018). A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106, 249–259.
- Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., ... & Beijbom, O. (2020). nuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 11621–11631). IEEE.
- Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 321–357.
- Chen, Y., Fu, M., & Shen, K. (2022). Point-BLS: 3D point cloud classification combining deep learning and broad learning system. In *2022 34th Chinese Control and Decision Conference (CCDC)* (pp. 2810–2815). IEEE.
- Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, 88(2), 303–338.
- Fan, L., Huang, J., Wu, S., Zhu, Y., & Wang, Y. (2022). Learning with LiDAR intensity: Improving 3D object detection and segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 23(8), 10422–10434.

- Geiger, A., Lenz, P., & Urtasun, R. (2013). Are we ready for autonomous driving? The KITTI vision benchmark suite. *International Journal of Robotics Research*, 32(9–10), 1231–1237.
- Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow* (2nd ed.). O'Reilly Media.
- Gong, Z., Kang, C., Wang, A., Wang, J., Hu, Y., & Lu, J.-G. (2021). 3D point clouds classification and segmentation network based on global features. In *2021 China Automation Congress (CAC)* (pp. 1344–1348). IEEE.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- Griffiths, D., & Boehm, J. (2019). A review on deep learning techniques for 3D sensed data classification. *Remote Sensing*, 11(12), 1499.
- Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., & Bennamoun, M. (2020). Deep learning for 3D point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12), 4338–4364.
- Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K., & Pollefeys, M. (2017). Semantic3D.net: A new large-scale point cloud classification benchmark. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-1-W1, 91–98.
- Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362.
- He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9), 1263–1284.
- He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)* (pp. 2980–2988). IEEE.
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95.
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning* (pp. 448–456). PMLR.
- Ioffe, S., & Szegedy, C. (2015). *Batch normalization: Accelerating deep network training by reducing internal covariate shift*. In *Proceedings of the 32nd International Conference on Machine Learning* (pp. 448–456). PMLR.
- Johnson, J. M., & Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1), 1–54.
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.
- Lisboa, L., Silva, L., Oliveira, R., & Neves, A. J. R. (2025). Multi-modal fusion of LiDAR and camera data for robust object detection in autonomous driving. *IEEE Access*, 13, 11892–11907.
- Liu, X., & Yan, X. (2021). Point cloud feature extraction and classification for 3D modeling of indoor environments using LiDAR data. *Remote Sensing*, 13(6), 1107.
- Louppe, G. (2014). Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502*.
- Minemura, K., Liao, H., Monrroy, A., & Kato, S. (2018). LMNet: Real-time multiclass object detection on CPU using 3D LiDAR. In *2018 3rd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)* (pp. 28–34). IEEE.
- Murphy, K. P. (2022). *Probabilistic machine learning: An introduction*. MIT Press.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)* (pp. 8024–8035).
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Peterson, L. E. (2009). K-nearest neighbor. *Scholarpedia*, 4(2), 1883. <https://doi.org/10.4249/scholarpedia.1883>
- Pineau, J., Vincent-Lamarre, P., Sinha, K., Larivière, V., Beygelzimer, A., d'Alché-Buc, F., ... & Harchaoui, Z. (2021). Improving reproducibility in machine learning research: A report from the NeurIPS 2019 reproducibility program. *Journal of Machine Learning Research*, 22(1), 1–20.
- Pirotti, F., Santoro, M., & Miele, S. (2020). Indoor object classification from LiDAR data using deep learning. *ISPRS Journal of Photogrammetry and Remote Sensing*, 162, 113–123.
- Pomerleau, F., Colas, F., Siegwart, R., & Magnenat, S. (2015). Comparing ICP variants on real-world data sets. *Autonomous Robots*, 34(3), 133–148.
- Powers, D. M. W. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1), 37–63.
- Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 652–660). IEEE.

- Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017). PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NeurIPS)* (pp. 5099–5108).
- Rao, M., Yuan, S., Tang, P., & Ge, J. (2022). Siamese-PointNet++: Point cloud classification with Siamese PointNet++. In *2022 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML)* (pp. 221–224). IEEE.
- Raschka, S., & Mirjalili, V. (2020). *Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2*. Packt Publishing.
- Reimers, N., & Gurevych, I. (2017). Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 338–348). Association for Computational Linguistics.
- Riou, K., Subrin, K., & Callet, P. L. (2022). Reinforcement learning based point-cloud acquisition and recognition using exploration-classification reward combination. In *2022 IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1–6). IEEE.
- Shan, J., & Toth, C. K. (Eds.). (2018). *Topographic laser ranging and scanning: Principles and processing*. CRC Press.
- Sievert, C. (2020). *Interactive web-based data visualization with R, plotly, and shiny*. Chapman & Hall/CRC.
- Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4), 427–437.
- Şen, A., & Bayaslı, B. (2021). Hava Lidar verilerinin denetimsiz yapay sinir ağları kullanılarak filtrelenmesi. *Journal of Geodesy and GeoInformation*, 8(1), 18–29.
- Velodyne. (2020). *HDL-32E datasheet*. <https://velodynelidar.com> adresinden alınmıştır.
- Wu, B., Zhou, X., Zhao, S., Yue, X., & Keutzer, K. (2019). SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud. In *2019 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 4376–4382). IEEE.
- Yin, T., Zhou, X., & Krähenbühl, P. (2021). Center-based 3D object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 11784–11793). IEEE.
- Zhang, K., Zhang, L., & Li, P. (2016). A review of LiDAR data classification methods for vegetation studies. *Remote Sensing*, 8(7), 621.
- Zhang, Y., & Rabbat, M. (2018). A graph-CNN for 3D point cloud classification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6279–6283). IEEE.

- Zhang, Y., Zhao, H., Jiang, H., & Liu, S. (2022). Efficient visualization and processing of large-scale point clouds using WebGL-based platforms. *Journal of Visual Communication and Image Representation*, 84, 103482.
- Zhang, Z. (2016). Introduction to machine learning: k-nearest neighbors. *Annals of Translational Medicine*, 4(11), 218.
- Zhou, Y., & Tuzel, O. (2018). VoxelNet: End-to-end learning for point cloud based 3D object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4490–4499). IEEE.



ÖZGEÇMİŞ

Nisa KOBAN, lise eğitimini Cumhuriyet Anadolu Lisesi'nde (2012-2015) ve Sınav Temel Lisesi'nde (2015-2016) tamamladı. 2017- 2018 öğretim yılında Gümüşhane Üniversitesi Elektrik Elektronik Mühendisliği lisans eğitimine başladı. 2021 yılında bu bölümden mezun oldu. 2022 yılında Karadeniz Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Elektrik-Elektronik Mühendisliği Anabilim dalında tezli yüksek lisans programına başladı. 2024 yılından itibaren Türk Telekom Trabzon Bölge Müdürlüğünde görev yapmaktadır.

Yayınlar

- a. ÇAVLI, N., Koban, N., BENLİ, E., & ATASOY, A., (2023). *Motion Prediction Classification for Wall-Following Robot Navigation* . 8th International Scientific Conference “Telecommunications, Informatics, Energy and Management” – TIEM 2023, Bandırma, Turkey
- b. Koban, N., Çavli, N., Doğan, H., & Benli, E., (2023). *7 DOF Robotic Arm Inverse Kinematic Analysis with Cuckoo and Whale Algorithms* . 2023 Innovations in Intelligent Systems and Applications Conference (ASYU) (pp.1-6). Sivas, Turkey