

EffiTrack*

*Note: Anomaly Detection and Optimization in High Rack Storage Systems

1nd Artun Kara

Bilişim Sistemleri Mühendisliği Bölümü
Kocaeli Üniversitesi
Kocaeli, Türkiye
211307030@kocaeli.edu.tr

2nd Esmagül Kılınçat

Bilişim Sistemleri Mühendisliği Bölümü
Kocaeli Üniversitesi
Kocaeli, Türkiye
211307025@kocaeli.edu.tr

Abstract—This study addresses the integration of big data analytics and machine learning techniques to enhance energy efficiency and detect anomalies in high-bay storage systems. Innovative approaches for real-time data processing and anomaly detection have been implemented using technologies such as Apache Kafka and Apache Spark. The study aims to resolve energy consumption imbalances through data preprocessing, classification, and optimization techniques. The results demonstrate improvements in energy consumption and increases in operational efficiency. Future studies will explore the scalability of the system and its applicability in various industrial domains.

Index Terms—Büyük Veri, Makine Öğrenimi, Anomali Tespiti, Enerji Verimliliği, Apache Spark, Apache Kafka, Yüksek Raf Depolama Sistemleri

I. GİRİŞ (INTRODUCTION)

Endüstri 4.0 devrimi, 2011 yılında Almanya'da üniversiteler, özel şirketler ve Alman Federal Hükümeti'nin iş birliği ile başlatılan stratejik bir girişimdir. Bu devrim, endüstriyel üretimin verimliliğini, etkinliğini ve üretkenliğini artırmayı hedefleyen modern bir üretim sistemini ortaya koymaktadır. Endüstri 4.0, geleneksel üretim süreçlerini dijitalleşme ile yeniden şekillendirmiş ve yenilikçi teknolojilerin entegrasyonu ile zeki fabrikalar gibi çığır açıcı konseptlerin doğmasına yol açmıştır.

Bu çerçevede, Endüstri 4.0'ın temel amacı, üretim süreçlerinde kullanılan makineleri ve diğer fiziksel sistemleri birbirirle iletişim kurabilecek şekilde yapılandırarak esnek ve özelleştirilebilir üretim süreçlerini mümkün kılmaktır. Bu süreç, Nesnelerin İnterneti (IoT), Büyük Veri ve Siber Fiziksel Sistemler gibi teknolojilerle desteklenmektedir. Böylelikle zeki fabrikalar, sadece üretim verimliliğini artırmakla kalmayıp aynı zamanda çevre dostu üretim süreçleri ve müşteri odaklı çözümler sunmayı hedeflemektedir.

Zeki fabrikalar, üretim süreçlerinin dijitalleşmesini esas alarak, küresel iş birliği ve gerçek zamanlı veri alışverişiyle çalışan sistemler sunar. Bu fabrikalar, dinamik ve hızlı değişen taleplere uyum sağlamak amacıyla esnek üretim hatları kullanır. Aynı zamanda, ürünlerin yaşam döngüsü boyunca minimum maliyetle maksimum verim sağlamak için karmaşık üretim sistemleri arasında optimize edilmiş süreçler yürütürler. Ancak bu karmaşıklık, sistemlerde bazı istenmeyen durumların, yani anomalilerin, oluşmasına da sebep olabilmektedir.

Anomaliler, zeki fabrikalarda birçok sorun yaratabilir. Örneğin, sistemde yaşanan bir anomali, üretim hattında gecikmelere, kaynak israfına, kalite düşüşüne ve hatta büyük maliyet kayıplarına neden olabilir. Bu nedenle, zeki fabrikaların en kritik ihtiyaçlarından biri, bu anomalileri önceden tespit edebilecek sistemlerin geliştirilmesidir. Anomali tespit sistemleri, zeki üretim süreçlerinde sürekli çalışan makinelerin veri akışını izler ve herhangi bir anormal davranış hızla belirleyerek operatörleri bilgilendirir. Bu, üretim hattının kesintisiz çalışmasını sağlamak ve potansiyel kayıpları en aza indirmek açısından hayatı önem taşır.

Bu bağlamda, makine öğrenimi algoritmaları, zeki fabrikalarda anomali tespiti için önemli bir araç haline gelmiştir. Örneğin, Kaggle platformunda yer alan ve yüksek raflı depolama sistemleri üzerine yapılan bir çalışmadan alınan veri setleri, bu tür algoritmaların uygulanması ve performanslarının ölçülmesi için kullanılmıştır. Çalışmada, makine öğrenimi algoritmaları, zeki fabrikalarda enerji optimizasyonu ve anomalilerin tespit edilmesi gibi konularda ne derece etkili olduklarını göstermek için analiz edilmiştir. Rastgele Orman, C4.5 Karar Ağacı ve k-En Yakın Komşu gibi algoritmaların kullanıldığı bu çalışmalar, algoritmaların anomali tespitindeki doğruluk oranlarını kıyaslamayı mümkün kılmıştır.

Sonuç olarak, bu çalışmalar, sadece zeki fabrikaların üretim süreçlerini optimize etmeye kalmamış, aynı zamanda bu süreçlerde yaşanabilecek problemleri önceden tespit ederek maliyetleri düşürme ve sistemlerin verimliliğini artırma fırsatlarını da ortaya koymustur. Bu yönyle, Endüstri 4.0'ın temel hedefleri olan sürdürülebilirlik, verimlilik ve yenilikçilik, zeki fabrikalar aracılığıyla somut bir şekilde hayatı geçirilmiştir.

A. Problem Alanı

HRSS, sürekli artan enerji taleplerine uyum sağlarken, enerji tüketimindeki dalgalandırmalar ve verimsizliklerle mücadele etmek zorundadır. Sistemin doğasında bulunan bu sorunlar, yalnızca enerji kaybına değil, aynı zamanda operasyonel kesintilere ve maliyet artışlarına yol açmaktadır. Bu bağlamda aşağıdaki temel problem alanları tanımlanabilir:

- Enerji Tüketimi ve İsraf:** Depolama sistemlerinin enerji tüketim dinamikleri, operasyonel verimlilik hedeflerine ulaşmada büyük bir engel teşkil etmektedir. Geleneksel

izleme yöntemleri, enerji kayıplarını doğru şekilde tespit edememekte ve israfın önlenmesini zorlaştırmaktadır.

- **Operasyonel Verimsizlik:** Yüksek hacimli operasyonel süreçlerde meydana gelen verimsizlikler, üretim ve lojistik hatlarının planlanması olumsuz etkilemektedir. Bu durum, zamanında teslimat hedeflerini tehdit etmekte ve müşteri memnuniyetini azaltmaktadır.

- **Gerçek Zamanlı Veri Analizi Eksikliği:** HRSS'nin ürettiği büyük veri setleri, geleneksel yöntemlerle işlenemeyecek kadar karmaşıktır. Bu nedenle, operasyonel karar alma süreçlerinde gecikmeler yaşanmakta ve sistemlerin esnekliği azalmaktadır.

- **Anomali Tespit ve Enerji Kaybı Yönetimi:** Operasyonel süreçlerde meydana gelen olağanüstü durumlar, enerji kaybına yol açabilmekte ve sistemlerin uzun vadeli performansını olumsuz etkilemektedir. Anomali tespitinde kullanılan mevcut yöntemler, yetersiz kalmaktadır.

B. Araştırma Problemi

Bu çalışma, HRSS'nin enerji tüketim verilerinin analiz edilmesi, operasyonel süreçlerdeki verimsizliklerin tespit edilmesi ve enerji kayıplarının önlenmesi için yenilikçi bir yaklaşım geliştirmeyi amaçlamaktadır. Veri odaklı yaklaşımlar ve makine öğrenimi teknikleri kullanılarak, enerji yönetimi ve operasyonel süreçlerde karşılaşılan zorluklara çözüm önerileri sunulacaktır. Bununla birlikte, Apache Kafka ve Apache Spark gibi büyük veri teknolojilerinin entegrasyonu ile gerçek zamanlı analiz kabiliyetlerinin artırılması hedeflenmektedir.

C. Amaç ve Kapsam

Bu proje, Yüksek Raf Depolama Sistemi (HRSS) içerisinde operasyonel süreçlerin enerji verimliliği ve performans açısından optimize edilmesini hedeflemektedir. Sensörlerden toplanan veri setleri kullanılarak anomali tespiti gerçekleştirilmekte, bu anomalilerin operasyonel süreçler üzerindeki etkileri analiz edilmekte ve sistem performansını artırmaya yönelik çözümler sunulmaktadır. Veri ön işleme, sınıflandırma ve optimizasyon tekniklerinin uygulandığı proje, enerji tüketiminde sapmaları azaltmayı, taşıyıcı hareketlerini optimize etmeyi ve sürdürülebilir bir depolama süreci oluşturmayı amaçlamaktadır.

1) **Amaç:** Bu projenin amacı, Yüksek Raf Depolama Sistemi (HRSS) içerisinde enerji tüketimi, taşıyıcı hareketleri ve operasyonel verimlilik açısından ortaya çıkan anormal durumların tespit edilmesi ve bu anomalilerin neden olduğu olumsuz etkilerin en aza indirilmesidir. Sistem performansını artırmak için sensör verileri üzerinde veri analitiği ve makine öğrenimi teknikleri uygulanarak, hem enerji verimliliğini artırmayı hem de operasyonel sürekliliği sağlamayı hedeflemektedir.

2) **Kapsam:** Çalışma kapsamında, enerji yönetimi süreçlerine dair aşağıdaki adımlar gerçekleştirılmıştır:

- Yüksek Raf Depolama Sistemi sensörlerinden alınan veri setlerinin analizi ve temizlenmesi.
- Taşıyıcıların mesafe, hız, güç ve zamanlama verilerindeki anomalilerin tespiti.

- Normal ve anormal durumlar arasında ayırım yapmak için veri ön işleme ve sınıflandırma yöntemlerinin uygulanması.
- Enerji tüketiminde sapmalara neden olan faktörlerin belirlenmesi ve optimize edilmesi.
- Anomali tespitine dayalı olarak operasyonel süreçlerin iyileştirilmesi ve sistemin genel performansının artırılması.
- Veri seti üzerinde keşifsel analizler ve görselleştirmeler yaparak depolama sisteminin enerji tüketim dinamiklerini anlamak,
- Makine öğrenimi yöntemleri kullanarak yüksek raf depolama sisteminde enerji verimliliğini artırmaya yönelik tahminsel modeller geliştirmek,
- Apache Spark kullanarak depolama sisteme ait operasyonel verilerin hızlı ve verimli bir şekilde işlenmesini sağlamak,
- Anomali tespit algoritmaları ile depolama sistemi süreçlerinde olağanüstü durumları belirlemek ve operasyonel kayıpları azaltmak,
- Yüksek raf depolama sistemlerinin yönetim süreçlerini veri odaklı karar alma ile optimize etmek.

Bu kapsamda gerçekleştirilen çalışmaların, enerji depolama sistemlerinde operasyonel verimliliği artırarak enerji maliyetlerini düşürmesi ve daha sürdürülebilir bir enerji yönetim sistemi geliştirilmesine katkı sunması hedeflenmektedir.

D. Problem Tanımı

Enerji yönetimi, günümüzün en önemli küresel sorunlarından biridir. Yüksek enerji talebi ve sınırlı enerji kaynakları, enerji sistemlerinde daha verimli ve sürdürülebilir çözümler geliştirmesini zorunlu kılmaktadır. Bu bağlamda, enerji depolama sistemleri, enerji talebi ve arzı arasındaki dengesizliği azaltmada kritik bir rol oynamaktadır. Ancak, bu sistemlerin etkin bir şekilde yönetilmesi, büyük miktarda verinin işlenmesini ve analiz edilmesini gerektirmektedir.

Bu çalışmada ele alınan problem, enerji depolama sistemlerinde:

- Operasyonel verimliliğin artırılmasını engelleyen bilinmezliklerin belirlenmesi,
- Enerji kullanımındaki anomali durumlarının tespiti ve nedenlerinin açıklanması,
- Gerçek zamanlı veri akışlarının işlenmesindeki teknik zorlukların üstesinden gelinmesi,
- Enerji optimizasyonu için makine öğrenimi ve büyük veri analitiği yöntemlerinin uygulanabilirliğinin değerlendirilmesi,

şeklinde tanımlanmaktadır.

Bu teknolojiler ve araçlar, proje kapsamında enerji verilerinin etkin bir şekilde analiz edilmesini ve işlenmesini sağlamak için entegre bir şekilde kullanılmıştır.

E. Kullanılan Teknolojiler ve Araçlar

Bu projede, enerji verilerinin işlenmesi, analizi ve görselleştirilmesi için aşağıdaki teknolojiler ve araçlar kullanılmıştır:

- Apache Kafka:** Gerçek zamanlı veri akışlarının yönetimi ve işlenmesi için kullanılan bir dağıtık yayın/abone platformu. Projede, enerji depolama sistemlerinden gelen büyük veri setlerinin akış tabanlı işlenmesi için kullanılmıştır.
- Apache Spark:** Büyük veri analitiği ve makine öğrenimi algoritmalarının hızlı bir şekilde çalıştırılması için kullanılan bir veri işleme motoru. Spark, verilerin dağıtık ortamda işlenmesi ve anomalî tespiti gibi işlemler için tercih edilmiştir.

Bu teknolojiler ve araçlar, proje kapsamında enerji verilerinin etkin bir şekilde analiz edilmesini ve işlenmesini sağlamak için entegre bir şekilde kullanılmıştır.

II. SİSTEMİN TANITIMI

A. Giriş

SmartFactoryOWL ve Almanya merkezli Institute Industrial IT iş birliği ile bir Yüksek Raf Depolama Sistemi (HRSS) demonstratörü geliştirilmiştir. Bu demonstratörün amacı, sisteme anomalileri analiz etmek ve teşhis etmektir. Veri seti, dört konveyör bandı (BLO, BHR, BHL, BRU) ve iki raydan (HL, HR) oluşan bir yüksek raflı depolama sistemi temel alınarak oluşturulmuştur. Sistem, iki nokta arasında paket taşıma amacıyla tasarlanmıştır. İlk veri seti, orta konveyör bantlarının yalnızca dikey hareket gerçekleştirdiği bir modda toplanmıştır. İkinci veri seti ise sistemin optimize edilmesiyle, orta bantların dikey hareket sırasında yatay yönlü paket taşıma işlemini de yapabildiği bir modda elde edilmiştir. Çalışmada, bu iki veri seti üzerinde sınıflama algoritmaları uygulanarak, optimizasyon işleminin etkileri incelenmiştir. Genel yaklaşım, aşağıdaki başlıklarda detaylandırılmıştır.

B. Fiziksel Akışın Anlaşılması

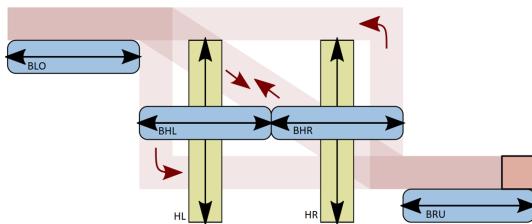


Fig. 1: Yüksek Raflı Depolama Sisteminin Görsel Modeli

HRSS demonstratörü, malların (veya nesnelerin) farklı raflar arasında taşınmasını sağlar. Sistemin çalışma prensipleri ve fiziksel akışı iki farklı iterasyonda incelemiştir:

- Standart İterasyon:** Sistem, temel çalışma prensipleri doğrultusunda bir "milk run" şeklinde çalışır.
- Optimize İterasyon:** Enerji optimizasyonunu simülé eden bir çalışma modeli uygulanmıştır.

Fiziksel akışın detaylı incelemesi, 4 adet konveyör (BRU, BHR, BHL, BLO) ve 2 adet ray sistemi (HR, HL) üzerine odaklanmıştır. Konveyörlerin her biri elektrik motorlarıyla çalışmaktadır ve kayışlar yardımıyla hareket etmektedir. Bu sistemler, yatay ve dikey taşıma görevlerini yerine getirmek için tasarlanmıştır.

1) *Standart Fiziksel Akış:* Standart iterasyonda, fiziksel akış şu şekilde gerçekleşmektedir:

- 1) **İleri Hareket:** Palet, BRU konveyörünün sağ kenarına yerleştirilir ve sırasıyla BRL, BHR, BHL ve BLO konveyörleri üzerinden sağdan sola hareket eder.

- 2) **Geri Hareket:** Palet, BLO konveyörünün sol kenarından sağ kenarına doğru hareket eder ve sırasıyla BHL, BHR ve BRU üzerinden başlangıç noktasına döner.

2) *Optimize Fiziksel Akış:* Optimize iterasyonda, enerji tasarrufu sağlamak amacıyla fiziksel akış daha verimli bir şekilde düzenlenmiştir:

- 1) **İleri Hareket:** Palet, BRU konveyöründen başlayarak, dikey raylar (HR ve HL) üzerinden optimize edilmiş bir rota ile BLO'ya taşınır.

- 2) **Geri Hareket:** Palet, BLO'dan başlayarak optimize edilmiş bir şekilde BRU'ya geri döner.

C. Veri Çıkarm Tekniği

Bu deneyin temel amacı, sistemin kendi kendini teşhis edebilmesini ve anomalileri tespit edebilmesini sağlamaktır. Bunun için, normal koşullarda toplanan veriler ile anomalili veriler karşılaştırılarak aşağıdaki analizler gerçekleştirilmiştir:

- Sistem davranış modellerinin çıkarılması,
- Anomalî türlerinin ve zamanlamalarının belirlenmesi,
- Enerji optimizasyonu performansının değerlendirilmesi.

Her bir konveyör, üç induksiyon sensörü ile donatılmıştır. Sensörlerin konumları şu şekildedir:

- İlk sensör, sol kenardan 36mm uzaklıkta,
- İkinci sensör, sol kenardan 266mm uzaklıkta,
- Üçüncü sensör, sağ kenardan 36mm uzaklıkta.

Ray sistemine ait sensör bilgisi bulunmamakla birlikte, BHR ve BHL konveyörlerindeki sensörlerin ray verilerini ölçmek için kullanılmış olabileceği varsayılmaktadır. Deney kapsamında iki veri dosyası sağlanmıştır:

- Standart Hareket,
- Optimize Hareket,

III. VERİ SETİ ÖN İŞLEME (DATASET PREPROCESSING)

Bu bölümde, *HRSS_normal_standard.csv* ve *HRSS_normal_optimized.csv* veri setleri üzerinde gerçekleştirilen ön işleme adımları açıklanmaktadır ve elde edilen sonuçlar sunulmaktadır. Spark ve Scala veris setinin anlaşılması için Python kullanılarak aşağıdaki işlemler gerçekleştirilmiştir.

A. Veri Setinin Yüklenmesi

Veri seti, Spark DataFrame API'si kullanılarak yüklenmiştir. Veri setlerinin şeması Tablo I'da verilmiştir.

Tablo I'da belirtilen veri seti sütunlarının açıklamaları aşağıda verilmiştir:

- **Timestamp:** Her bir veri kaydının zaman damgasını temsil eder. Veriler, milisaniye cinsinden kaydedilmiştir.
- **Labels:** Her bir verinin sınıf etiketini belirtir. Örneğin, 0 normal veriyi, 1 ise anomalili veriyi ifade eder.

TABLE I: Veri Seti Şeması

Sütun Adı	Veri Tipi	Eksik Veri (Nullable)
Timestamp	double	true
Labels	integer	true
I_w_BLO_Weg	double	true
O_w_BLO_power	double	true
O_w_BLO_voltage	double	true
I_w_BHL_Weg	double	true
O_w_BHL_power	double	true
O_w_BHL_voltage	double	true
I_w_BHR_Weg	double	true
O_w_BHR_power	double	true
O_w_BHR_voltage	double	true
I_w_BRU_Weg	double	true
O_w_BRU_power	double	true
O_w_BRU_voltage	double	true
I_w_HR_Weg	double	true
O_w_HR_power	double	true
O_w_HR_voltage	double	true
I_w_HL_Weg	double	true
O_w_HL_power	double	true
O_w_HL_voltage	double	true

- **I_w_BLO_Weg:** BLO (bileşen) için giriş akımının ölçüm değerini ifade eder.
- **O_w_BLO_power:** BLO bileşeni için çıkış gücü (Watt) ölçüm değerini ifade eder.
- **O_w_BLO_voltage:** BLO bileşeni için çıkış voltajı (Volt) ölçüm değerini ifade eder.
- **I_w_BHL_Weg:** BHL bileşeni için giriş akımının ölçüm değerini ifade eder.
- **O_w_BHL_power:** BHL bileşeni için çıkış gücü ölçüm değerini ifade eder.
- **O_w_BHL_voltage:** BHL bileşeni için çıkış voltajı ölçüm değerini ifade eder.
- **I_w_BHR_Weg:** BHR bileşeni için giriş akımının ölçüm değerini ifade eder.
- **O_w_BHR_power:** BHR bileşeni için çıkış gücü ölçüm değerini ifade eder.
- **O_w_BHR_voltage:** BHR bileşeni için çıkış voltajı ölçüm değerini ifade eder.
- **I_w_BRU_Weg:** BRU bileşeni için giriş akımının ölçüm değerini ifade eder.
- **O_w_BRU_power:** BRU bileşeni için çıkış gücü ölçüm değerini ifade eder.
- **O_w_BRU_voltage:** BRU bileşeni için çıkış voltajı ölçüm değerini ifade eder.
- **I_w_HR_Weg:** HR bileşeni için giriş akımının ölçüm değerini ifade eder.
- **O_w_HR_power:** HR bileşeni için çıkış gücü ölçüm değerini ifade eder.
- **O_w_HR_voltage:** HR bileşeni için çıkış voltajı ölçüm değerini ifade eder.
- **I_w_HL_Weg:** HL bileşeni için giriş akımının ölçüm değerini ifade eder.
- **O_w_HL_power:** HL bileşeni için çıkış gücü ölçüm değerini ifade eder.
- **O_w_HL_voltage:** HL bileşeni için çıkış voltajı ölçüm değerini ifade eder.

Veri setinin ilk 5 satırı Tablo II'de gösterilmektedir.

TABLE II: Veri Setinin İlk 5 Satırı

Timestamp	Labels	I_w_BLO_Weg	O_w_BLO_power	O_w_BLO_voltage	...
0.0	0	-107.0	0.0	0.0	...
0.0459976196289063	0	-107.0	0.0	0.0	...
0.1510009765625	0	-107.0	0.0	0.0	...
0.206001281738281	0	-107.0	0.0	0.0	...
0.263999938964844	0	-107.0	0.0	0.0	...

B. Eksik Veri Analizi

Veri setinde eksik değer bulunup bulunmadığını kontrol etmek amacıyla her bir sütundaki null değerlerin sayısı hesaplanmıştır. Sonuçlara göre, veri setinde hiçbir sütunda eksik değer bulunmamaktadır:

- Null Değer Sayısı: 0

C. Ön İşlem Öncesi Veri Görselleştirme

Veri görselleştirme, veri setinin yapısını anlamak ve modelleme öncesinde gerekli işlemleri belirlemek için kritik bir adımdır. Bu adımda, veri setindeki önemli sütunların dağılımları ve ilişkileri incelenmiştir.

1) *Sayısal Sütunların Histogramları:* Bu adımda, veri setindeki sayısal sütunların dağılımlarını anlamak, sütunlar arasındaki ilişkileri belirlemek ve eksik değerleri kontrol etmek amacıyla sayısal veri analizi gerçekleştirilmiştir. Bu işlemler için kullanılan kod, numerical-data-analysis.py isimli bir Python betiğinde hazırlanmıştır.. Analiz sonuçları aşağıda detaylı bir şekilde açıklanmıştır.

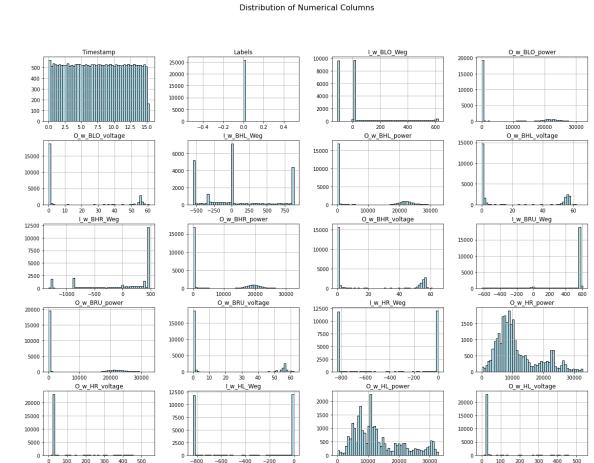


Fig. 2: Normal veri setindeki histogramlar, birçok değişkenin belirli aralıklarda yoğunlaştığıını ancak bazı sütunlarda anomal uç değerlerin bulunduğuunu göstermektedir.

a) *Standard Veri Seti:* Standard veri setinde gerçekleştirilen histogram analizleri, verilerin geniş bir varyans sergilediğini ve belirli özelliklerin farklı aralıklarda yoğunlaştığını göstermiştir. Bu durum, operasyonel süreçlerde belirgin dengesizliklerin bulunduğu ve bazı değişkenlerin sistemin enerji tüketim dinamiklerinde önemli dalgalanmalara yol açabileceğini ortaya koymaktadır. Bu tür dağılımlar, hem

süreç optimizasyonu hem de anomalilerin tespiti için detaylı bir şekilde incelenmelidir.

- Güç (Power) sütununda birden fazla tepe noktası (multi-modal) tespit edilmiştir. Bu durum, farklı yük koşullarında çalışıldığını ve enerji tüketiminde tutarsızlıkların olduğunu göstermektedir.
- Hız (Speed) sütununda uzun kuyruklar gözlemlenmiştir. Bu durum, belirli süreçlerin aşırı hızda veya düşük hızda gerçekleştiğini ve potansiyel anomali kaynaklarını işaret etmektedir.
- Zamanlama (Timing) ve mesafe (Distance) sütunlarında dağınık bir dağılım söz konusudur. Bu, süreçlerin düzenli bir şekilde işlemeyebileceğini göstermektedir.

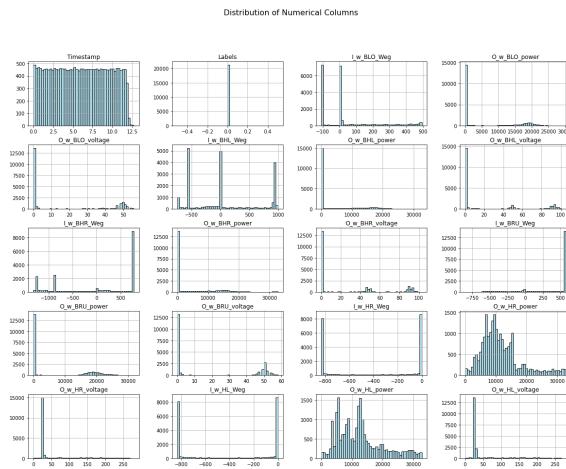


Fig. 3: Optimize veri setindeki histogramlar, operasyonel süreçlerin daha düzenli ve dar aralıklarda yoğunlaştığı, anomal uç değerlerin ise minimize edildiğini göstermektedir.

b) *Optimized Veri Seti:* Optimized veri setinde histogramlar daha dar aralıklarda yoğunlaşmış olup, süreçlerin daha tutarlı ve optimize edilmiş bir şekilde çalıştığını göstermektedir.

- Güç (Power) sütununda tek modlu (tek tepe noktası) bir dağılım gözlemlenmiştir. Bu, enerji tüketiminde daha düşük varyans ve dengeli bir tüketim profili anlamına gelir.
- Hız (Speed) sütununda veriler daha dar aralıkta yoğunlaşmış olup, taşıyıcıların hız kontrolünün iyileştirildiğini ve süreçlerdeki varyansın azaltıldığını göstermektedir.
- Mesafe (Distance) ve zamanlama (Timing) sütunlarında daha düzenli ve odaklanmış bir dağılım gözlemlenmiştir. Bu, operasyonel süreçlerin kontrol altına alındığını ve enerji tüketimindeki sapmaların minimize edildiğini göstermektedir.

2) Korelasyon Matrisi:

a) Normal Veri Seti:

- **Genel Korelasyon Yapısı:** Normal veri setindeki korelasyon matrisi, değişkenler arasında karmaşık bir ilişkili ağına işaret etmektedir. Güç (Power) ve gerilim (Voltage) gibi değişkenler arasında belirli pozitif korelasyonlar gözlemlenmiştir.

• **Güçlü Pozitif Korelasyonlar:**

- $O_w_BHR_power$ ve $O_w_BHR_voltage$ arasında güçlü bir pozitif korelasyon bulunmaktadır. Bu, enerji tüketimi ile gerilim değişkenlerinin paralel hareket ettiğini göstermektedir.
- Benzer şekilde, $O_w_BRU_power$ ve $O_w_BRU_voltage$ arasında da güçlü bir ilişki görülmektedir.

- **Zayıf veya Negatif Korelasyonlar:** Bazı değişkenler arasında zayıf veya negatif korelasyonlar gözlemlenmiştir. Bu durum, sistemdeki belirli süreçlerin birbirinden bağımsız olduğunu veya ters etkilendirdiğini göstermektedir.

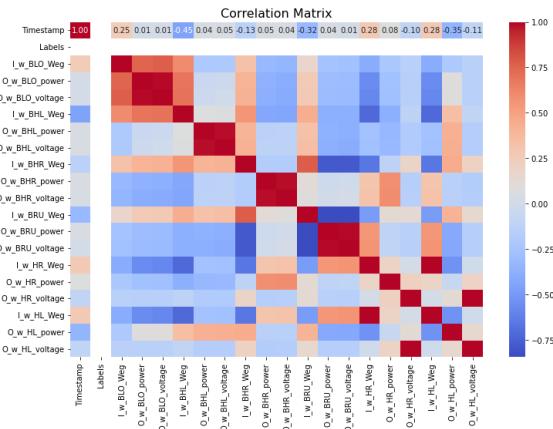


Fig. 4: Normal veri setindeki korelasyon matrisi, güç ve gerilim değişkenleri arasında yüksek pozitif korelasyonlar olduğunu, bazı değişkenler arasında ise zayıf veya negatif ilişkiler bulunduğu ortaya koymaktadır.

b) Optimize Veri Seti:

- **Genel Korelasyon Yapısı:** Optimize veri setindeki korelasyon matrisi, normal veri setine kıyasla daha net ve düzenli bir ilişki yapısı ortaya koymaktadır. Özellikle, güç (Power) ve gerilim (Voltage) değişkenleri arasındaki korelasyonlar belirginleşmiştir.

• Güçlü Pozitif Korelasyonlar:

- Optimize veri setinde, $O_w_HL_power$ ve $O_w_HL_voltage$ arasında oldukça güçlü bir pozitif korelasyon gözlemlenmiştir. Bu, optimizasyonun süreçler üzerindeki tutarlılığını artırdığını göstermektedir.

- **Negatif Korelasyonların Azalması:** Normal veri setinde bulunan bazı negatif korelasyonlar optimize veri setinde minimize edilmiştir. Bu durum, sistemin daha uyumlu bir şekilde çalıştığını ve operasyonel dengesizliklerin azaltıldığını ifade eder.

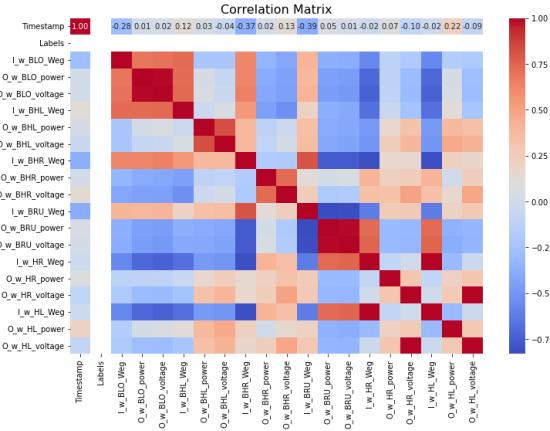


Fig. 5: Optimize veri setindeki korelasyon matrisi, güç ve gerilim değişkenleri arasında güçlü pozitif korelasyonların olduğunu ve süreç uyumunun arttığını ortaya koymaktadır.

3) Eksik Veri Analizi: Veri setinde eksik değer olmamadığını belirlemek için bir analiz gerçekleştirilmiştir. Yapılan incelemeler sonucunda veri setinde herhangi bir eksik değer bulunmadığı tespit edilmiştir. Bu durum, ön işleme aşamasını kolaylaştırmakta, eksik değer doldurma veya çıkarma gibi ek işlemleri gereksiz kılmaktadır.

4) Sonuç: Bu çalışma kapsamında, numerical-data-analysis.py adlı Python betiği kullanılarak veri seti üzerinde ön işlem öncesi çeşitli analizler ve görselleştirmeler gerçekleştirilmiştir.

Son olarak, eksik veri analizi gerçekleştirilmiş ve veri setinde eksik değer bulunmadığı görülmüştür. Bu durum, veri temizleme aşamasını basitleştirmış ve ön işleme sürecini hızlandırmıştır.

Elde edilen bulgular, veri setinin genel yapısını anlamak ve sonraki işleme adımlarını planlamak için değerli bilgiler sunmuştur. Özellikle, sınıf dengesizliği, aykırı değerler ve korelasyon gibi noktalar, modelleme sürecinde dikkate alınacaktır.

D. Eksik Verilerin Giderilmesi

Veri setinde yapılan analizler sonucunda eksik verilere rastlanmamıştır. Bu durum, veri temizleme aşamasını önemli ölçüde basitleştirmiştir. Eksik veri bulunmaması, modelleme sürecine geçmeden önce herhangi bir veri imputation (doldurma) veya çıkarma işlemi yapmayı gereksiz kılmaktadır. Bu nedenle, eksik veri ile ilgili herhangi bir ön işleme adımı uygulanmamıştır.

E. Veri Tipi Dönüşümleri

Veri setindeki mevcut sütunlar incelendiğinde, veri tipi dönüşümüne ihtiyaç duyulmadığı anlaşılmıştır. Zaman damgası (Timestamp) ve etiket sütunu (Labels) zaten uygun veri tiplerinde sunulmuş olup, sayısal sütunlar da doğru formatta yer almaktadır. Bu nedenle, veri tipi dönüşümleri gerçekleştirilmemiştir.

Veri setindeki sayısal sütunlar, analiz ve modelleme için uygun veri tiplerinde olup, tarihsel veriler de sayısal formatta saklanmaktadır. Labels sütunu, zaten sınıflandırma problemleri için uygun olan 0 ve 1 etiketleriyle temsil edilmiştir. Bu durum, ek bir veri tipi dönüşümü işlemeye gerek kalmadan verinin doğrudan kullanılabileceği anlamına gelmektedir.

Sonuç olarak, bu adımda herhangi bir veri tipi dönüşümü yapılmadan, veri seti doğrudan sonraki adımlara geçmek için hazır durumdadır.

F. Özelliğin Seçimi ve Özelliğin Çıkarımı

Veri setlerinden anlamlı özellikler seçmek ve çıkartmak, fiziksel akışın doğrudan gözlemlenememesi nedeniyle zorluklar içermektedir. Bu çalışmada kullanılan HRSS demonstratörü, malların farklı raflar arasında taşınmasını sağlayan bir sistemdir. Sistem, iki farklı iterasyonda (standart ve optimize) çalıştırılmış olup, fiziksel akışın enerji verimliliği ve operasyonel davranışını üzerine odaklanılmıştır. Bu bağlamda, özellik seçimi ve özellik çıkarımı konularına doğrudan müdahale edilememiştir; çünkü sistemin yapısı, önceden tanımlanmış fiziksel bileşenler ve veri akışı üzerinden çalışmaktadır.

a) Veri Seti Bağlamında Özellik Analizi: HRSS demonstratöründen elde edilen veriler, sistemin işleyişini doğrudan yansitan fiziksel ölçümlerden oluşmaktadır. Bu ölçüler, zaman damgası, sensör çıkış değerleri, akım, güç ve voltaj gibi özellikler içermektedir. Sistem, belirli bir fiziksel akışa bağlı çalıştığından, özellikler arasında çıkarım yapılabilecek bir veri yapısı bulunmamaktadır. Bunun yerine, mevcut özellikler kullanılarak sistem davranışlarının ve anomalilerinin analizi gerçekleştirilmiştir.

b) Fiziksel Akışın Veriye Yansımı: Sistemin fiziksel akışı, dört konveyör (BRU, BHR, BHL, BLO) ve iki ray sistemi (HR, HL) üzerinden sağlanmaktadır. Her bir konveyör, elektrik motorları ve sensörler yardımıyla hareketi düzenlemekte ve ölçümler yapmaktadır. Sensörlerden elde edilen veriler, sistemin fiziksel durumunu ve enerji tüketimini analiz etmek için kullanılmaktadır. Bu nedenle, mevcut veriler üzerinde yapılabilecek işlemler, öncelikli olarak sistem davranış modellerini çıkarmaya ve anomalileri tespit etmeye yönelikir.

c) Veri Setinde Özellik Seçimi ve Çıkarımı Üzerine Kısıtlar: Veri seti, doğrudan fiziksel ölçümlerden oluşduğundan, özellik seçimi ve çıkarımı için aşağıdaki kısıtlarla karşılaşmıştır:

- **Bağlam Bağımlılığı:** Sistem verileri, yalnızca belirli bir fiziksel yapı ve akış üzerinden toplandığı için yeni özellikler türetmek mümkün değildir.
- **Özelliklerin Fiziksel Anlamı:** Veri setindeki tüm özellikler, sistemin işleyişini birebir yansitan fiziksel anımlara sahiptir. Örneğin, voltaj ve akım ölçümleri enerji tüketimini analiz etmek için kullanılmaktadır.
- **Kapsamlı Birleştirme İhtiyacı:** Özellik çıkarımı için, farklı sensörlerden gelen verilerin birleştirilmesi gerekmektedir. Ancak bu işlem, sistemin mevcut yapısını değiştirmeyi gerektirebilir.

G. Kategorik Verilerin Dönüşürtürülmesi

Veri setinde, kategorik verilerin dönüştürülmesi işlemi genellikle metinsel veya sınıflandırılmış kategorik verilerin sayısal forma çevrilmesi amacıyla uygulanır. Ancak bu işlem, mevcut veri setinde aşağıdaki nedenlerle gerçekleştirilememektedir:

- Veri Türlerinin Tamamının Sayısal Olması:** HRSS veri seti, fiziksel sistemlerden gelen ölçüm verilerini içermektedir. Bu veriler, akım, voltaj, güç gibi sürekli (sayısal) değerlerden oluşmaktadır. Dolayısıyla, metin tabanlı veya sınıflandırılmış kategorik veri bulunmamaktadır. Örneğin, `Timestamp` sütunu zaman bilgisi sunarken, diğer sütunlar sistem bileşenlerinden gelen sensör verilerini temsil etmektedir.
- Kategorik Veri Eksikliği:** Veri setinde, kategorik olarak tanımlanabilecek sütunlar yer almamaktadır. Kategorik veri eksikliği nedeniyle, bu tür verilerin sayısal bir forma dönüştürülmesi için gereken işlemler uygulanamamaktadır. Kategorik veri dönüşümü gerektiren bir örnek, renk veya sınıf gibi metinsel veriler içerirdi. Ancak, mevcut veri seti yalnızca fiziksel ölçüm verilerinden oluşmaktadır.
- Sistemin Fiziksel Yapısına Uygunluk:** HRSS sisteminin çalışma prensipleri ve veri setindeki sütunlar, doğrudan fiziksel ölçüm sonuçlarını yansıtmaktadır. Bu nedenle, kategorik bir sınıflandırma yapısına ihtiyaç duyulmamış, dolayısıyla kategorik verilerin dönüştürülmesi adımı gereksiz hale gelmiştir.

Sonuç olarak, HRSS veri setinin yapısı ve içeriği, kategorik verilerin dönüştürülmesi adımlının uygulanabilirliğini sınırlamaktadır. Bu durum, veri setinin doğasına uygun olmayan işlemlerden kaçınılmamasını sağlamış ve yalnızca ilgili ön işleme adımlarının uygulanmasına odaklanılmıştır.

H. Özelliğin Ölçeklenmesi Feature Scaling

Özelliğin Ölçeklenmesi (*Feature Scaling*), makine öğrenimi ve veri analizi süreçlerinde farklı ölçekteki özelliklerin dengelenmesi amacıyla kullanılan temel bir ön işleme adımıdır. Ancak, HRSS veri setinde bu işlem aşağıdaki nedenlerden dolayı gerçekleştirilememektedir:

- Veri Özelliklerinin Homojenliği:** HRSS veri seti, yalnızca sensörlerden gelen fiziksel ölçüm değerlerini (akım, voltaj, güç) içermektedir. Bu değerler, aynı fiziksel ölçüm sistemine dayandığı için zaten benzer bir ölçekte bulunmaktadır. Dolayısıyla, özellik ölçeklenmesi işlemine ihtiyaç duyulmamaktadır.
- Mutlak Değerlerin Anlamı:** Veri setindeki mutlak değerler (örneğin, akım ve voltaj) sistemin enerji tüketimi ve performans analizi için kritik öneme sahiptir. Bu değerlerin ölçeklenmesi, analiz sürecinde anlam kaybına yol açabilir. Özellikle, enerji optimizasyonu gibi çalışmalar için bu değerlerin doğrudan kullanılması gerekmektedir.
- Fiziksel Akışa Bağlılık:** HRSS sisteminde fiziksel akışın modellenmesi, sensörlerden elde edilen ham verilerin

doğrudan analizi üzerine kuruludur. Bu nedenle, veriler üzerinde ölçekleme gibi dönüşümler yapmak, sistemin gerçek davranışını yansıtmayan sonuçlara yol açabilir.

- Özellikler Arası Doğrudan Kiyaslama Gerekliği Olmaması:** Veri setindeki özellikler, farklı sistem bileşenlerinden (BRU, BHR, BHL, BLO, HR ve HL) gelen ölçümleri temsil etmektedir. Bu bileşenler arasında doğrudan kıyaslama yapılması amaçlanmadığı için, ölçekleme işlemi gereksiz hale gelmiştir.

Sonuç olarak, HRSS veri setinin yapısı, özellik ölçekleme işlemini gereksiz kılmaktadır. Verilerin homojen yapısı, fiziksel ölçümlerin doğrudan anlamlı olması ve analiz sürecinin fiziksel akışa bağlı olması, bu işlemin uygulanamamasının temel nedenleri arasında yer almaktadır.

I. Uç Değerlerin İşlenmesi (Outliers Processing)

1) Anomali Tespiti İçin Kullanılan Yöntemler:

- a) Meksika Şapkası Dalgacık Yöntemi:** Meksika Şapkası Dalgacı yöntemi, her bir ölçümün referans değere olan mesafesini hesaplayarak, belirlenen eşik değerini aşan verileri anomali olarak işaretler. Bu çalışmada eşik değeri, anomali oranlarını optimize etmek amacıyla %60 olarak belirlenmiştir. Çok düşük eşik değerleri normal verilerin yanlışlıkla anomali olarak işaretlenmesine yol açarken, çok yüksek eşik değerleri gerçek anomalilerin gözden kaçmasına neden olabilmektedir.

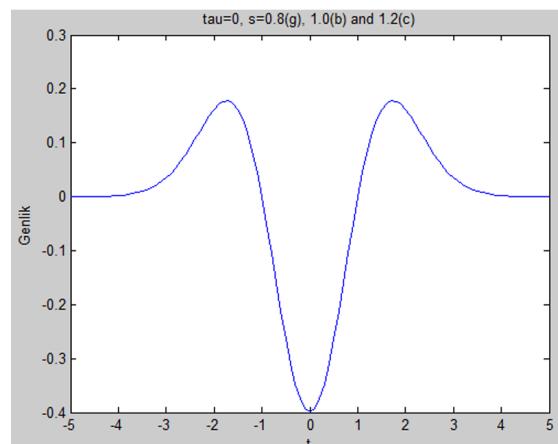


Fig. 6: Meksika Şapkası Dalgacık Grafiği

Anomali tespit işlemleri aşağıdaki adımları içermektedir:

- Meksika Şapkası Dalgacık Hesaplaması:** Her bir veri noktası için dalgacık değeri, aşağıdaki matematiksel ifade kullanılarak hesaplanmıştır:

$$\psi(x) = \frac{1}{\sqrt{3}\sigma\pi^{\frac{1}{4}}} \left(1 - \frac{x^2}{\sigma^2} \right) e^{-\frac{x^2}{2\sigma^2}} \quad (1)$$

Burada, σ parametresi Gauss fonksiyonunun standart sapmasını temsil etmektedir ve bu çalışmada $\sigma = 1.0$ olarak alınmıştır.

- Mesafe ve Eşik Değer Kontrolü:** Dalgacık değeri belirli bir eşik değeri (0.6) aşan veri noktaları anomali olarak işaretlenmiştir.

3) **Etiketleme (Labeling):** Anomali olarak işaretlenen veriler için *Labels* sütunu güncellenmiş ve anomali durumu 1 olarak atanmıştır. Normal veriler için ise 0 değeri korunmuştur.

b) *Kod ve Uygulama Detayları:* Aşağıda, anomalilerin tespiti ve işaretlenmesi için kullanılan kodun ilgili kısmından fonksiyonların açıklamaları verilmiştir:

- `calculateAnomalyUdf` fonksiyonu, her bir veri noktası için dalgacık değerini hesaplar ve anomali eşik değerini aşanları 1 olarak işaretler.
- `columnsToCheck` değişkeni, `Timestamp` sütunu hariç tüm ölçüm sütunlarını içermektedir.
- Etiketleme işlemi, `Labels` sütununda gerçekleştirilmiştir.

2) *Z-Skoru ile Uç Değerlerin Tespiti:* Z-Skoru yöntemi, veri noktalarının ortalamanadan kaç standart sapma uzaklıkta olduğunu hesaplayarak uç değerleri tespit eder. Z-Skoru aşağıdaki denklemle tanımlanır:

$$Z = \frac{X - \mu}{\sigma} \quad (2)$$

Burada:

- X : Veri noktası,
- μ : Veri setinin ortalaması,
- σ : Veri setinin standart sapması.

Bu yöntemde, $|Z| > 3$ olan değerler uç değer olarak kabul edilmiştir. Z-Skoru yöntemi ile *Labels* sütunu hariç tüm sütunlar incelenmiş ve uç değerler veri setinden temizlenmiştir.

a) *Kod ve Uygulama Detayları:* Aşağıda, uç değerlerin tespiti ve temizlenmesi için kullanılan kodun ilgili kısmından fonksiyonların açıklamaları verilmiştir:

- `detectOutliersZScore` fonksiyonu, her sütun için ortalama ve standart sapmayı kullanarak uç değerleri tespit etmektedir.
- `removeOutliers` fonksiyonu, tespit edilen uç değerleri veri setinden kaldırmaktadır.
- `cleanedOptimizedDF` ve `cleanedStandardDF`, optimize ve standart veri setlerindeki temizlenmiş veriyi temsil etmektedir.

TABLE III: Veri Seti Özeti

Veri Seti	Toplam Satır Sayısı	Anomali Sayısı (<i>Labels</i> = 1)
Standart Veri Seti	23645	5670
Optimize Veri Seti	19634	4517

3) *Veri Seti Özeti ve Sonuçlar:* Sonuçlar, her iki yöntemin de başarıyla uygulandığını ve veri setindeki uç değerlerin tespit edilip temizlendiğini göstermektedir. Meksika Şapkası Dalgacık yöntemi ile *Labels* sütununda anomaliler işaretlenirken, Z-Skoru yöntemi ile veri seti daha temiz bir analiz için optimize edilmiştir.

J. Veri dengesizliğinin giderilmesi

Bu çalışmada, enerji tüketimi verilerinde yer alan dengesiz sınıf dağılımı problemi, daha etkili bir makine öğrenimi modeli geliştirebilmek için ele alınmıştır. Veri dengesizliği, sınıflar

arasındaki örnek sayılarındaki önemli farklar nedeniyle, modelin nadir görülen sınıfı (örneğin, anomali verileri) öğrenmekte zorlanmasına yol açabilir. Bu sorunun üstesinden gelmek için iki temel yaklaşım uygulanmıştır: Undersampling ve SMOTE (Synthetic Minority Oversampling Technique).

1) *Undersampling:* Bu çalışmada, veri dengesizliğini gidermek amacıyla sınıflar arası dengeyi sağlamak için **Undersampling** yöntemi kullanılmıştır. Undersampling, veri setinde çoğunluk sınıfına ait örneklerin sayısını azaltarak azınlık sınıfıyla dengeli hale getirilmesini sağlar. Bu işlem, modelin çoğunluk sınıfına odaklanması engelleyerek azınlık sınıfını daha iyi öğrenmesini hedefler.

a) *İşlemenin Adımları::* Undersampling işlemi sırasında aşağıdaki adımlar izlenmiştir:

- 1) **Veri Setinin Yüklenmesi:** Veri seti Apache Spark kullanılarak yüklenmiştir. Veri setinin örnek satırları ve şeması incelenerek sütun isimleri ve veri tipleri doğrulanmıştır.
- 2) **Sınıf Dağılımının Analizi:** Veri setinde bulunan sınıfların dağılımı `groupBy` ve `count` fonksiyonları kullanılarak analiz edilmiştir. Bu analiz sonucunda çoğunluk ve azınlık sınıfları belirlenmiştir.
- 3) **Çoğunluk Sınıfının Azaltılması:** Çoğunluk sınıfına ait örnekler, azınlık sınıfına ait örnek sayısına uygun bir oran (*sampling fraction*) kullanılarak rastgele seçilmiştir. Bu işlem `sample` fonksiyonu ile gerçekleştirilmiştir.
- 4) **Veri Setinin Dengelenmesi:** Azınlık sınıfına ait tüm örnekler ile azaltılmış çoğunluk sınıfı birleştirilerek dengeli bir veri seti oluşturulmuştur.
- 5) **Sonuçların Kaydedilmesi:** Dengeli veri seti birleştirilmiş ve CSV formatında kaydedilmiştir. Kaydetme işlemi sırasında veri tek bir dosyada toplanacak şekilde `coalesce(1)` kullanılmıştır.

b) *Sonuçlar:* Undersampling işlemi sonucunda veri setindeki sınıf dağılımı dengelenmiş ve her iki sınıf için aşağıdaki örnek sayıları elde edilmiştir:

- **Optimize Yaklaşımı:**
 - Sınıf 1 (anomali): 4,517 adet
 - Sınıf 0 (normal): 4,496 adet
- **Standart Yaklaşımı:**
 - Sınıf 1 (anomali): 5,670 adet
 - Sınıf 0 (normal): 5,706 adet

Bu işlem sonucunda, sınıflar arası dengesizlik azaltılmış ve makine öğrenimi modellerinin performansını artıracak bir veri seti elde edilmiştir.

2) *SMOTE (Synthetic Minority Oversampling Technique):* SMOTE, veri setlerindeki dengesiz sınıf dağılımlarını gidermek için kullanılan etkili bir teknik olup, azınlık sınıfına ait sentetik veri örnekleri üreterek sınıf dengesini sağlar. Bu teknik, azınlık sınıfına ait mevcut veri noktaları arasındaki komşuluk ilişkilerini kullanarak yeni veri noktaları oluşturur ve böylece veri setindeki azınlık sınıfının temsil gücünü artırır. Bu çalışmada, SMOTE işlemi Apache Spark kullanılarak gerçekleştirilmiştir.

a) *İşlemin Adımları*: SMOTE işlemi sırasında aşağıdaki adımlar izlenmiştir:

- 1) **Veri Setinin Yüklenmesi**: eri seti Apache Spark kullanılarak yüklenmiştir. Veri setindeki tüm özellikler (*features*) VectorAssemblers ile tek bir vektör haline getirilmiştir.
- 2) **Azınlık ve Çoğunluk Sınıflarının Belirlenmesi**: Veri setindeki sınıf dağılımı analiz edilerek azınlık (*minority*) ve çoğunluk (*majority*) sınıfları tespit edilmiştir.
- 3) **K-En Yakın Komşuluk Hesaplaması**: Her bir azınlık sınıfı örneği için *K-En Yakın Komşular* (*K-Nearest Neighbors*, $k=5$) algoritması uygulanmıştır.
- 4) **Sentetik Örneklerin Üretilmesi**: Rastgele bir azınlık sınıfı örneği ile komşularından biri arasındaki vektörel mesafeye bağlı olarak yeni veri noktaları (*synthetic samples*) oluşturulmuştur. Bu işlem, veri noktaları arasındaki öklid mesafesi temel alınarak gerçekleştirılmıştır.
- 5) **Veri Setinin Dengelenmesi**: Üretilen sentetik azınlık örnekleri, azınlık sınıfı ve çoğunluk sınıfı verileriyle bireleştirilerek dengeli bir veri seti elde edilmiştir.
- 6) **Sonuçların Kaydedilmesi**: Dengeli veri seti CSV formatında dışa aktarılmıştır.

b) *Sonuçlar*: SMOTE işlemi sonucunda, sınıflar arası dengesizlik giderilmiş ve veri seti aşağıdaki şekilde dengelenmiştir:

- **Standart Yaklaşım**:

- Sınıf 1 (anomali): 17,975 adet
- Sınıf 0 (normal): 17,975 adet

- **Optimize Yaklaşım**:

- Sınıf 1 (anomali): 15,117 adet
- Sınıf 0 (normal): 15,117 adet

Bu işlemler sonucunda dengeli veri seti elde edilerek, makine öğrenimi modellerinin azınlık sınıfını daha iyi öğrenmesi sağlanmıştır. SMOTE işleminin detayları, uygulama sırasında kullanılan parametreler ve üretilen veri örnekleri ile birlikte, model performansına etkisi ilerleyen bölümlerde tartışılacaktır.

3) *Neden Veri Dengesizliği Giderildi?*: Veri dengesizliği, makine öğrenimi modellerinin performansı üzerinde önemli olumsuz etkiler yaratır. Dengesiz veri setlerinde, çoğunluk sınıfına ait örnekler azınlık sınıfına göre daha fazla olduğundan, model eğitimi sırasında çoğunluk sınıfını öğrenmeye daha fazla ağırlık verir. Bu durum, özellikle azınlık sınıfına ait verilerin model tarafından doğru bir şekilde sınıflandırılması zorlaştırır.

Bu çalışmada, enerji tüketim verilerinde anomalilerin tespit edilmesi kritik öneme sahiptir. Ancak, anomali verilerinin nadiren gözlemlenmesi nedeniyle veri setinde ciddi bir sınıf dengesizliği bulunmaktadır. Modelin azınlık sınıfına (anomali) ait verileri doğru bir şekilde öğrenmemesi, enerji yönetimi ve verimlilik analizlerinde yanlış sonuçlara yol açabilir. Bu nedenle, veri setindeki sınıf dağılımını dengelemek, şu nedenlerle hayatı öneme sahiptir:

- **Anomalilerin Daha İyi Tespiti**: Azınlık sınıfını temsil eden anomali verilerinin tespiti, enerji kayıplarının

önlenmesi ve operasyonel süreçlerdeki aksaklılıkların belirlenmesi açısından kritik öneme sahiptir.

- **Model Performansının Artırılması**: Dengesiz veri setleri, modelin azınlık sınıfına ait örneklerde düşük hassasiyet (*recall*) ve düşük doğruluk (*accuracy*) göstermesine neden olabilir. Veri dengesizliğinin giderilmesi, modelin genelleme yeteneğini artırır ve tüm sınıflarda daha dengeli bir performans sağlar.
- **Önyargının Azaltılması**: Dengesiz veri setleri, modelin çoğunluk sınıfına önyargı geliştirmesine neden olabilir. Dengeli veri seti, modelin tüm sınıfları eşit şekilde öğrenmesini teşvik eder.
- **Gerçek Dünya Problemlerine Uygunluk**: Özellikle enerji yönetimi ve arıza tespiti gibi gerçek dünya problemlerinde, azınlık sınıfı genellikle en kritik bilgiye sahiptir. Bu nedenle, modelin azınlık sınıfını etkili bir şekilde öğrenmesi gereklidir.

Yukarıda belirtilen nedenlerle, veri dengesizliğini gidermek amacıyla **Undersampling** ve **SMOTE (Synthetic Minority Oversampling Technique)** yöntemleri uygulanmıştır. Bu yöntemler, veri setinin sınıf dağılımını dengelerken, veri kaybını en aza indirgemek ve model performansını artırmak için tasarlanmıştır.

4) *Sonuçlar*: Bu çalışmada uygulanan Undersampling ve SMOTE işlemleri sonucunda, veri setindeki sınıf dengesizliği başarılı bir şekilde giderilmiştir. Aşağıda bu işlemlerin elde ettiği sonuçlar özeti verilmiştir:

- **Undersampling Sonuçları**:

- Optimize Yaklaşımı:
 - * Sınıf 1 (anomali): 4,517 adet
 - * Sınıf 0 (normal): 4,496 adet
- Standart Yaklaşımı:
 - * Sınıf 1 (anomali): 5,670 adet
 - * Sınıf 0 (normal): 5,706 adet

- **SMOTE Sonuçları**:

- Standart Yaklaşımı:
 - * Sınıf 1 (anomali): 17,975 adet
 - * Sınıf 0 (normal): 17,975 adet
- Optimize Yaklaşımı:
 - * Sınıf 1 (anomali): 15,117 adet
 - * Sınıf 0 (normal): 15,117 adet

Bu işlemler sonucunda elde edilen dengeli veri setleri, makine öğrenimi modellerinin eğitim ve test süreçlerinde kullanılmıştır. Dengeli veri setleri, modelin azınlık sınıfını daha iyi öğrenmesini sağlamış, böylece anomali tespit performansı önemli ölçüde artırılmıştır. Ayrıca, sınıf dengesizliğinin giderilmesi ile elde edilen sonuçların, enerji yönetimi süreçlerinde verimliliğin artırılması ve enerji kayıplarının önlenmesine yönelik önemli katkılar sunduğu görülmüştür.

5) *Sonraki Adımlar*: Dengeli veri setleri üzerinde eğitilen modellerin performansı, ilerleyen bölgelerde detaylı bir şekilde analiz edilecektir. Bu analizlerde, modellerin doğruluk (*accuracy*), hassasiyet (*precision*), hatırlama (*recall*) ve F1 skoru gibi performans metrikleri üzerinden değerlendirilmeleri

yapılacaktır. Ayrıca, enerji yönetimi süreçlerinde uygulanabilirlik ve verimlilik üzerine olan etkileri tartışılacaktır.

K. Vektörleştirme

Vektörleştirme, genellikle metin verisi veya kategorik verileri sayısal formata dönüştürmek için kullanılan bir tekniktir. Ancak, bu çalışmada kullanılan veri seti tamamen sayısal ölçümlerden (*akım, voltaj, güç* gibi)oluştuğu için vektörleştirme işlemi gereksiz ve uygulanamaz. Bunun başlıca nedenleri şunlardır:

- 1) **Sayısal Verilerden Oluşması:** Veri setindeki tüm özellikler sayısal olduğu için makine öğrenimi algoritmaları bu verileri doğrudan işleyebilir.
- 2) **Kategorik veya Metin Verinin Bulunmaması:** Veri setinde herhangi bir kategorik değişken veya metin verisi bulunmamaktadır.
- 3) **Doğrudan Kullanılabilirlik:** Sayısal ölçümler doğrudan analiz ve modelleme için uygun olduğundan, vektörleştirme gibi ara işlemlere ihtiyaç yoktur.

L. Örneklemme (Sampling)

Örneklemme, genellikle büyük veri setlerinden bir alt küme seçilerek analiz yapmayı ifade eder. Ancak, bu çalışmada kullanılan veri seti (*HRSS_normal_standard* ve *HRSS_normal_optimized*) üzerinde örneklemme işlemi uygulanmamıştır. Bunun başlıca nedenleri aşağıda açıklanmıştır:

- 1) **Veri Setinin Boyutu:** Veri seti boyutu orta düzeyde olduğu için, tüm veri üzerinde işlem yapmak makul bir sürede gerçekleştirilebilir. Bu durumda örneklemme işlemine gerek yoktur.
- 2) **Doğu Temsil Gereksinimi:** Anomaliler ve uç değerler gibi nadir olaylar, örneklemme sırasında gözden kaçabilir. Bu durum, analiz sonuçlarının doğruluğunu etkileyebilir.
- 3) **Homojen Dağılım Varsayıtı:** Örneklemme, veri setinin homojen bir dağılıma sahip olduğunu varsayar. Ancak, fiziksel sistemlerden elde edilen bu veri seti, farklı durumları ve dağılımları içerebilir.
- 4) **Verinin Özelliği:** Veri seti tamamen fiziksel ölçümlerden oluşmaktadır ve bu ölçümlerin tamamı analizin doğruluğu için gereklidir.

Sonuç olarak, örneklemme işlemi bu veri setinde uygulanmamış ve tüm veri seti analize dahil edilmiştir. Bu yaklaşım, nadir olayların gözden kaçmasını engellemekte ve analiz sonuçlarının doğruluğunu artırmaktadır.

IV. GÖRSELLEŞTİRME (VISUALIZATION)

Bu bölümde, HRSS demonstratörünün fiziksel akışı ve enerji optimizasyon süreçlerine dair veri setleri görselleştirilerek analiz edilmiştir. Analiz, aşağıdaki unsurları kapsayacak şekilde tasarlanmıştır:

- Veri setinin genel yapısının tanımlanması ve özetlenmesi,
- Dağılım ve trend analizlerinin histogramlar ve scatter plot gibi temel görselleştirme yöntemleriyle incelenmesi,
- Değişkenler arasındaki ilişkilerin ve korelasyonların heatmap gibi görsellerle sunulması,

- Standart ve optimize iterasyonlara dair farkların görsellerle ifade edilmesi.

Verilerin görselleştirilmesi ve analiz edilmesi, hem sistemin mevcut durumunu anlamayı hem de potansiyel anomalilerin tespit edilmesini kolaylaştırmaktadır. Bu analizlerle elde edilen içgörüler, enerji optimizasyonunun etkilerini değerlendirme ve sistem performansını iyileştirme açısından kritik öneme sahiptir.

Veri setlerindeki önemli noktaları anlamak için histogram, scatter plot ve heatmap gibi çeşitli veri görselleştirme yöntemleri kullanılmıştır. Bu bölümde, standart ve optimize iterasyonları arasındaki farkları ortaya çıkarmak ve enerji tüketimi üzerine odaklanılmıştır.

A. Histogram Analizi

Yapılan histogram analizi, her iki iterasyondaki enerji tüketimlerini karşılaştırmak için kullanılmıştır. Histogramlar, veri setlerindeki değerlerin dağılımını ve olası anomalileri görüntülemek için etkili bir yöntemdir. Bu analiz, standart ve optimize iterasyonların enerji tüketim profillerini anlamamıza yardımcı olur.

Yapılan histogram analizi, her iki iterasyondaki enerji tüketimlerini karşılaştırmak için kullanılmıştır. Şu sütunlar göz önüne alınmıştır:

- O_w_BLO_power
- O_w_BHL_power
- O_w_BHR_power
- O_w_BRU_power
- O_w_HR_power
- O_w_HL_power

Bu sütunlarda histogramlar oluşturulmuş ve optimize ile standart iterasyonların enerji tüketim dağılımları özelliklerine göre karşılaştırılmıştır.

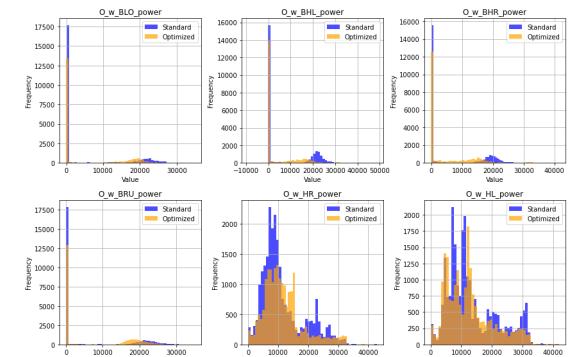


Fig. 7: Oluşturulan histogramların optimize ile standart iterasyonları

1) Görülebilen Bulgular:

- Optimize iterasyonda enerji tüketiminin daha dengeli bir dağılım sergilediği gözlemlenmiştir.
- Standart iterasyonda belirli sütunlarda enerji tüketiminin çok yüksek değerlerde yoğunluğu fark edilmiştir.
- Optimize iterasyon enerji verimliliğine odaklandığı için genel olarak daha düşük enerji tüketimi gözlenmiştir.

B. Label Dağılım Analizi

Bu analiz, sistemin optimize iterasyonda daha az anomali oluşturduğunu ve bunun enerji verimliliği üzerindeki etkilerini göstermek amacıyla gerçekleştirilmiştir. Anomalilerin azaltılması, sistemin daha stabil bir performans sergilediğini ve optimize iterasyonun faydalarnı ortaya koyar.

Standart ve optimize veri setlerinde bulunan anomali durumu etiketlerini incelemek için şu adımlar uygulanmıştır:

- Labels sütununun frekansları analiz edilmiştir.
- Normal ("0") ve anomalik ("1") durumların dağılımları karşılaştırılmıştır.

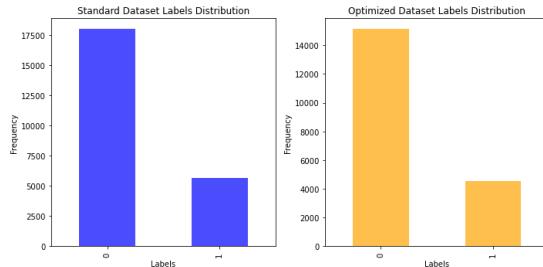


Fig. 8: Anomalilerin analizi

1) Görülebilen Bulgular:

- Standart veri setinde 15.117 normal durum, 4.517 anomalik tespit edilmiştir.
- Optimize veri setinde ise daha az sayıda anomalik gözlemlenmiştir, bu da sistemdeki optimizasyonun anomalik oranını azalttığını işaret etmektedir.

C. Korelasyon Analizi: Standart ve Optimize Veri Setleri

Korelasyon analizi, enerji tüketimi ile ilgili değişkenler arasındaki ilişkileri anlamak ve sistem optimizasyonun etkilerini değerlendirmek amacıyla gerçekleştirilmiştir. Bu analizde standart ve optimize veri setleri arasındaki korelasyon farkları incelenmiş ve aşağıdaki sonuçlar elde edilmiştir.

1) Analizin Amacı:

- Enerji tüketimiyle ilgili bileşenler arasında nasıl bir ilişki olduğunu tespit etmek.
- Optimize veri setinde enerji verimliliği nedeniyle korelasyonun değişip değişmediğini değerlendirmek.
- Sistem optimizasyonun enerji tüketimi bileşenleri arasındaki bağımlılık ve düzenlilik üzerindeki etkisini analiz etmek.

2) Görülebilen Bulgular:

a) Standart Veri Seti::

- Standart veri setindeki korelasyon haritasında, bazı değişken çiftleri arasında yüksek pozitif ya da negatif korelasyon gözlemlenmiştir.
- Bununla birlikte, enerji tüketimindeki bazı bileşenler arasındaki ilişkiler düzensiz bir yapı sergilemiştir.
- Örneğin, $O_w_BLO_power$ ve $O_w_BHL_power$ arasında güçlü bir pozitif ilişki, diğer bileşenler arasında ise zayıf ilişkiler gözlemlenmiştir.

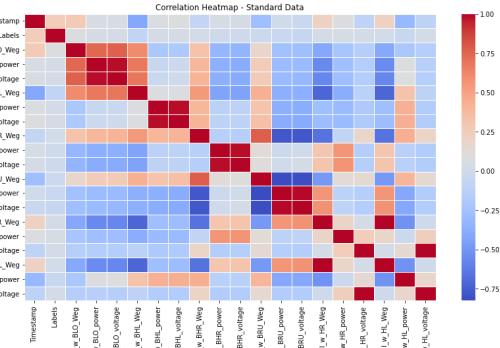


Fig. 9: Standart Veri Seti

b) Optimize Veri Seti:

- Optimize veri setinde korelasyon haritası, enerji tüketimi bileşenleri arasında daha düzenli ve belirgin bir ilişki göstermektedir.
- Bazı bileşenler arasındaki korelasyonların, optimizasyon sonrası güçlendiği veya zayıfladığı fark edilmiştir.
- Bu durum, optimizasyon enerji tüketimindeki dağılım ve bileşenler arasındaki etkileşimleri düzenlediğini göstermektedir.

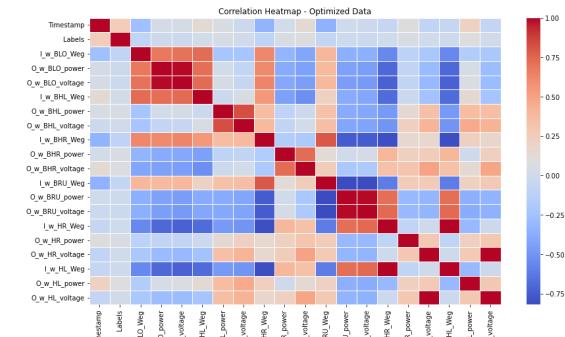


Fig. 10: Optimize Veri Seti

3) Sonuçlar ve Değerlendirme: Korelasyon analizinden elde edilen sonuçlar, enerji tüketiminde optimize iterasyonun standart iterasyona kıyasla daha düzenli ve dengeli bir yapı sağladığını ortaya koymuştur. Bu, optimize sisteme bileşenler arasındaki ilişkilerin daha sıkı ve öngörelebilir olduğunu ve genel enerji verimliliğinin artırıldığını işaret etmektedir.

D. Zaman Serisi Görselleştirme Sonuçları (Örneklenmiş Veri)

Zaman serisi görselleştirme, standart ve optimize veri setlerinde anomali etiketlerinin zaman içerisindeki dağılımını incelemek için kullanılmıştır. Bu analiz, anomalilerin zaman içerisindeki düzenliliğini ve optimizasyonun etkisini değerlendirmeyi amaçlamaktadır.

1) Standart Veri Seti:

- Anomalili verilerin zaman içerisindeki dağılımı düzensizdir.

- Anomaliler genellikle belirli zaman aralıklarında kümelenmiş görünüyor.

2) Optimize Veri Seti:

- Optimize veri setinde anomaliler daha homojen bir şekilde dağıtılmıştır.
- Anomalilerin sıklığı belirli dönemler mevcut olup, bu dönemde enerji tüketimi daha dikkatli analiz edilmelidir.

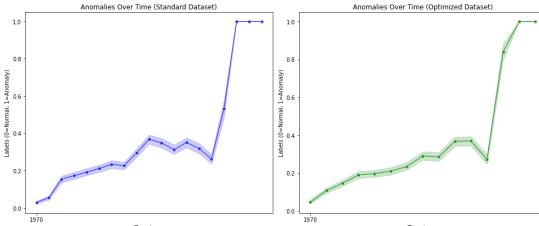


Fig. 11: Zaman Serisi Görselleştirme

3) Sonuçlar ve Değerlendirme: Zaman serisi görselleştirmeler, optimize veri setinde anomalilerin zaman içerisindeki dağılımının daha düzenli olduğunu ve bu durumun enerji yönetimindeki iyileştirmelerden kaynaklandığını göstermektedir. Standart veri setindeki kümelenmeler, sistemdeki potansiyel sorunlu dönemleri işaret ederken, optimize veri setindeki düzenlilik enerji verimliliği açısından olumlu bir gelişme olarak değerlendirilebilir.

E. Enerji Tüketimi Analizi Sonuçları

Enerji tüketimi analizi, standart ve optimize veri setlerindeki güç tüketim değerlerini anlamak ve iki veri seti arasındaki farklılıklar değerlendirmek amacıyla gerçekleştirilmiştir. Bu analiz, dağılım ve zaman trendleri açısından incelenmiştir.

1) Standart Veri Seti:

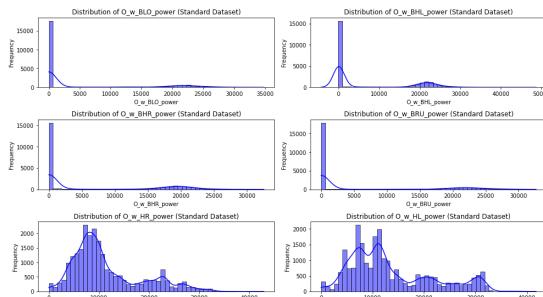


Fig. 12: Dağılımlar Standart Veri Seti

a) Dağılımlar:

- Güç (power) ile ilgili sütunların çoğu geniş bir dağılıma sahiptir.
- Bazı sütunlarda sıfır değerlerinin yoğunluğu yüksektir (örneğin, $O_w_BLO_power$), diğer sütunlarda değerler geniş bir aralığa yayılmıştır.
- Belirli sütunlarda enerji tüketiminde anormal derecede yüksek değerler gözlemlenmiştir.

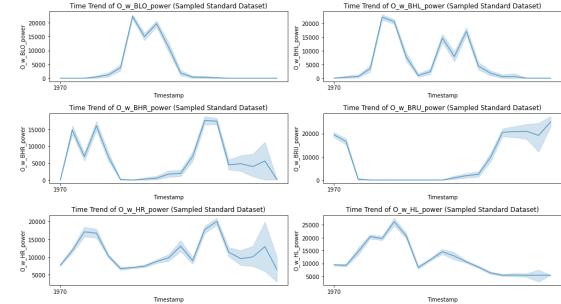


Fig. 13: Zaman Trendleri Standart Veri Seti

b) Zaman Trendleri:

- Güç tüketimi zamanla dalgalanmalar göstermektedir.
- Bazı sütunlarda enerji tüketiminde ani yükselmeler veya düşüşler dikkat çekmektedir.
- Bu düzensizlikler, standart sistemin enerji tüketiminde optimize edilmemiş bir yapıyı işaret etmektedir.

2) Optimize Veri Seti:

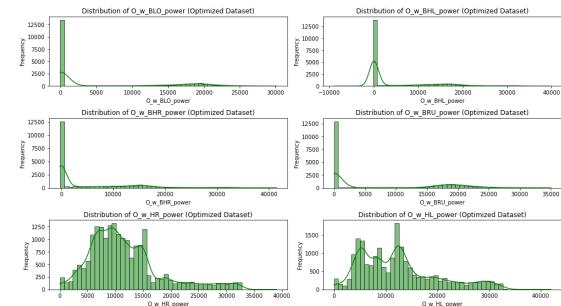


Fig. 14: Dağılımlar Optimize Veri Seti

a) Dağılımlar:

- Optimize veri setindeki güç (power) değerlerinin dağılımları, Standart Veri Seti'ne kıyasla daha dar bir aralıktan yoğunlaşmıştır.
- Sıfır değerlerinin hâlâ belirgin olduğu sütunlar mevcut olmakla birlikte, bazı bileşenlerde pozitif değerlerin yoğunluğu daha yüksektir.
- Bu durum, optimizasyonun enerji tüketim değerlerini dengelediğini göstermektedir.

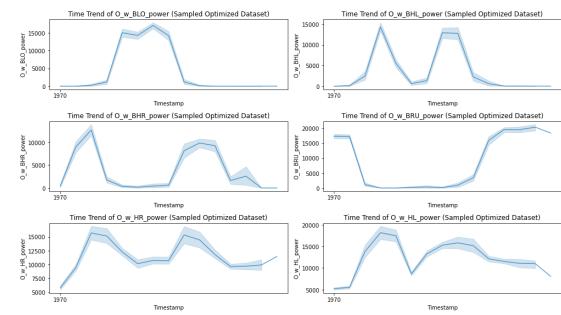


Fig. 15: Zaman Trendleri Optimize Veri Seti

b) Zaman Trendleri:

- Zaman içerisindeki enerji tüketimi daha düzenli bir yapı sergilemektedir.
- Bazı sütunlarda enerji tüketiminde belirgin artış veya azalış trendleri gözlemlenmiştir.
- Bu, optimize edilmiş süreçlerin etkisini ve enerji verimliliğinin artışını göstermektedir.

3) Sonuçlar ve Değerlendirme: Enerji tüketimi analizi, optimize veri setinin enerji tüketimindeki dengesizliği azaltarak daha verimli bir yapı sağladığını ortaya koymaktadır. Standart veri setindeki düzensizlikler, sistemdeki sorunlu süreçler işaret ederken, optimize veri setindeki düzenlilik, süreçlerin enerji verimliliğini artıran bir şekilde yeniden yapılandırıldığını göstermektedir.

F. Güç Bileşenlerinin Enerji Tüketimi Analizi (Violin Plot)

Bu bölümde, standart ve optimize veri setlerinde güç bileşenlerinin (BLO, BHL, BHR, BRU, HR, HL) enerji tüketimi, normal ve anomalili durumlar için *violin plot* görselleştirmeleri kullanılarak incelenmiştir. Analiz, enerji tüketimindeki farklılıklarları ve optimizasyonun etkilerini değerlendirmektedir.

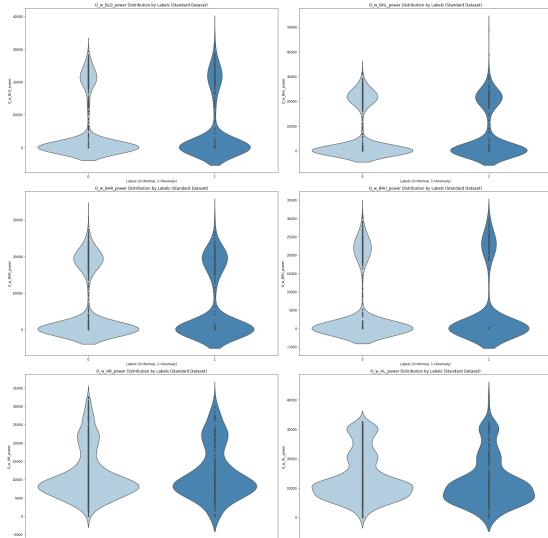


Fig. 16: Normal Veri Seti İçin Güç Bileşenlerinin Enerji Tüketimi Analizi (Violin Plot)

1) Standart Veri Seti:

a) Normal Veriler (Label = 0):

- BHR bileşeni dahil tüm bileşenlerde çıkış gücü genelde düşük ila orta seviyelerde yoğunlaşmıştır.
- Ancak, bazı durumlarda yüksek enerji tüketimi değerleri gözlemlenmiştir, bu da sistemdeki kontrolsüzlüklerin varlığına işaret etmektedir.

b) Anomalili Veriler (Label = 1):

- Güç dağılımları geniş bir aralıkta dağılmıştır ve yüksek enerji tüketimleri anomali durumları ile ilişkilendirilmiştir.
- Bileşenlerin çoğu enerji tüketiminde belirgin artışlar tespit edilmiştir.

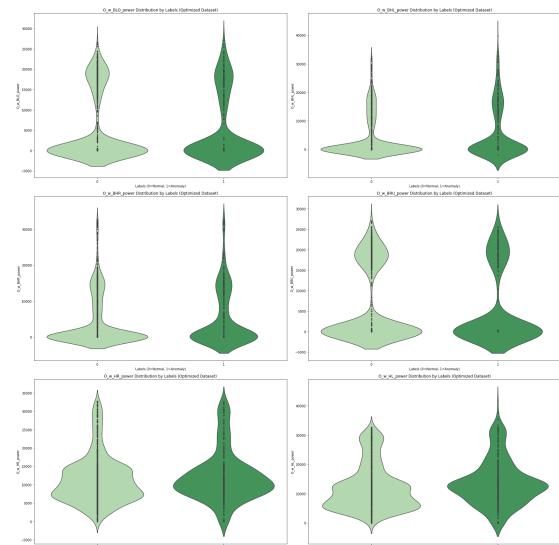


Fig. 17: Optimize Veri Seti İçin Güç Bileşenlerinin Enerji Tüketimi Analizi (Violin Plot)

2) Optimize Veri Seti:

a) Normal Veriler (Label = 0):

- Güç dağılımı daha dar bir aralıktaki yoğunlaşmış ve enerji tüketimi daha düzenli bir şekilde gerçekleşmiştir.
- Bu durum, optimizasyonun enerji tüketimini dengelediğini göstermektedir.

b) Anomalili Veriler (Label = 1):

- Anomalili durumlarda dahi enerji tüketimi daha dengeli bir şekilde gerçekleşmiştir.
- Yüksek enerji tüketimi değerlerinin sıklığı azalmış ve sistem daha kontrol edilebilir bir hale gelmiştir.

3) Sonuçlar Ve Amaç:

- Standart sisteme anomali durumlar enerji tüketiminde kontrollsüz artışlarla ilişkilendirilmiştir.
- Optimize sisteme, enerji tüketimi hem normal hem de anomalili durumlarda daha dar bir aralıktaki yoğunlaşmıştır.
- Normal ve anomalili durumların violin plot görselleştirmeleri, enerji tüketimindeki farklılıkları açıkça ortaya koymaktadır.

a) Son Değerlendirme: Optimize edilmiş sistemde enerji tüketiminin daha düzenli bir yapıya kavuştuğu, violin plot görselleştirmeleri ile açık bir şekilde görülmektedir. Bu durum, optimizasyon sürecinin sistemin enerji verimliliği üzerindeki olumlu etkilerini doğrulamaktadır.

G. Sonuç

Bu analizler, enerji tüketimindeki düzensizlikleri anlamak, optimizasyonun etkilerini değerlendirmek ve sistem performansını iyileştirme potansiyelini ortaya koymak için yapılmıştır. Veri setindeki bileşenler arasındaki enerji tüketim ilişkilerini anlamak, anomali tespiti ve enerji yönetimi stratejilerinin geliştirilmesi açısından kritik öneme sahiptir.

1) Amaçlar:

a) 1. Anomali ve Normal Durumları Karşılaştırma:

- Standart ve optimize veri setlerindeki anomali ve normal durumların enerji tüketimine nasıl yansığını analiz etmek.
- Anomalili durumların enerji tüketiminde nasıl bir denge-sizlik yarattığını incelemek.
- Normal ve anomalili durumlar arasındaki farkları ortaya koyarak sistemdeki enerji yönetimi sorunlarını tespit etmek.

b) 2. Optimizasyonun Enerji Yönetimine Etkisini Değerlendirme:

- Optimizasyonun enerji tüketim dağılımını nasıl etkilediğini anlamak.
- Anomalili durumlarda enerji tüketiminin nasıl kontrol altına alındığını değerlendirmek.
- Optimizasyon sonrası normal durumlarda daha verimli enerji tüketimi elde edilip edilmediğini incelemek.

c) 3. Sistem Performansını İyileştirme Potansiyelini Belirleme:

- Standart sistemdeki enerji tüketim düzensizliklerini tespit ederek, iyileştirme için kritik noktaları belirlemek.
- Optimizasyon hangi bileşenlerde daha fazla fayda sağladığını belirlemek.
- Gelecekteki enerji yönetimi stratejilerine rehberlik edecek veriler elde etmek.

2) Analizin Katkıları:

a) 1. Enerji Tüketimi Yönetimi:

- Standart sistemde enerji tüketiminde görülen düzensizlikler, sistemde kontrollsüz güç tüketiminin potansiyel kaynaklarını işaret etmektedir. Bu bilgiler, enerji tüketim yönetimini iyileştirmek için kritik öneme sahiptir.
- Optimizasyon sonrası enerji tüketiminde gözlemlenen düzenlilik, optimizasyon stratejilerinin başarısını ve sistem performansına olumlu etkilerini kanıtlamaktadır.

b) 2. Anomali Tespit ve Önleme:

- Anomalili durumlarda enerji tüketimindeki artışın minimize edilmesi, sistem güvenilirliğini ve enerji verimliliğini artırmak için kritik bir adımdır.
- Standart veri setindeki anomalilerin enerji tüketimindeki kontrollsüz artışlarla ilişkili olması, gelecekte anomali önleme stratejileri geliştirilmesine ışık tutmaktadır.

c) 3. Karar Destek Mekanizmaları:

- Bu analizler, sistem optimizasyonu ve enerji yönetimi alanında veri odaklı kararların alınmasını desteklemektedir.
- Anomalili durumların enerji tüketimi üzerindeki etkilerini anlamak, sistem tasarımları ve süreç iyileştirmeleri için kritik bilgiler sunmaktadır.

3) Sonuç Olarak: Yapılan bu analizler, enerji tüketimi ve sistem performansı üzerindeki etkileri daha derinlemesine anlamamızı olanak tanımıştır. Standart sistemdeki sorunlu alanlar net bir şekilde tespit edilirken, optimize sistemin

enerji verimliliğini artırdığı ve anomalili durumlarda enerji tüketimini dengelediği görülmüştür. Bu bilgiler, enerji yönetimi süreçlerini geliştirmek ve gelecekte daha sürdürülebilir sistemler tasarlamak için temel oluşturmaktadır.

V. KULLANILAN MODEL VE PERFORMANS ANALİZİ (MODEL AND PERFORMANCE ANALYSIS)

Bu bölümde, HRSS'nin enerji tmde, HRSS'nin enerji tüketim verilerinin analiz edilmesi için kullanılan teorik kavramlar ve modeller detaylandırılmaktadır. Makine öğrenimi ve derin öğrenme algoritmalarının yanı sıra, Transformer tabanlı yaklaşımalar da ele alınarak bu tekniklerin enerji verilerindeki örüntülerini yakalama ve çözüm önerileri sunmadaki etkinliği değerlendirilmiştir. değerlendirilmiştir.

A. Teorik Kavramlar

Bu bölümde, HRSS'nin enerji tüketim verilerinin analiz edilmesi için kullanılan teorik kavramlar ve modeller detaylandırılmaktadır. Makine öğrenimi ve derin öğrenme algoritmalarının yanı sıra, Transformer tabanlı yaklaşımalar da ele alınarak her bir modelin teoriye dayalı açıklaması ve enerji analizine katkıları ayrıntılı bir şekilde ele alınmıştır.

1) *Naive Bayes*: Naive Bayes (NB) algoritması, istatistiksel değerlere dayalı olarak çalışan ve sınıflandırma problemleri için sıkça kullanılan bir yöntemdir. Dinamik sistemlerde uygulanırken, sürekli olarak hesaplama işlemlerinin yapılmasını gerektirir. Bu algoritma, Bayes kuralı ile karar ağaçları modelinin birleştirilmesiyle geliştirilmiştir. NB, verilen bir örneğin her bir sınıfa ait olma olasılığını hesaplamak için Bayes kuralını kullanır.

Makine öğrenmesi uygulamalarında yaygın olarak kullanılan NB, koşullu olasılık hesaplamaları üzerine kuruludur ve Bayes kuralının en basit hali olarak görülür. Bu algoritma, örnek verilerin hangi sınıfa ait olduğunu bilindiği durumlarda kullanılır ve özellikle metin sınıflandırma işlemlerinde başarılı sonuçlar verdiği bilinmektedir.

NB algoritmasının temelinde, koşullu sınıf bağımsızlığı varsayımları bulunur. Bu varsayıma göre, bir niteliğin değeri, diğer niteliklerin değerlerinden bağımsız olarak ele alınır. Pratikte nitelikler arasında bağımlılıklar olsa da, bu varsayımlar hesaplamaların kolaylaştırılması amacıyla uygulanır. Eğer bu varsayımlar doğru kabul edilirse, NB algoritması diğer yöntemlere göre daha iyi sonuçlar verebilir. Ayrıca, bu algoritmda tüm niteliklerin sınıf tahmininde eşit derecede önemli olduğu varsayıılır.

NB teknigi, genellikle bir veri dizisindeki her bir verinin belirli bir sınıfa ait olma olasılığını hesaplamak için kullanılır. Bu olasılık, Bayes teoremine dayanarak maksimum seviyeye çıkarılır ve en yüksek olasılık değerine sahip sınıf, örneğin ait olduğu sınıf olarak belirlenir.

Özetle Naive Bayes, olasılık temelli bir sınıflandırma algoritması olup Bayes teoremi temelinde çalışmaktadır. Bu algoritma, her bir özelliğin diğerlerinden bağımsız olduğunu varsayar ve bu basit varsayımla nedeniyle "naive" olarak adlandırılır.

Enerji verilerinin analizi bağlamında, Naive Bayes algoritması çoğunlukla kategorik sınıflandırma problemlerinde kullanılabilir. Örneğin, enerji tüketimi seviyelerinin (düşük, orta, yüksek) tahmin edilmesi gibi durumlarda etkili bir çözüm sunar. Naive Bayes'in özellikle hızlı ve hesaplama maliyeti düşük bir algoritma olması, büyük veri setlerinde tercih edilmesini sağlar.

2) Logistic Regression: Lojistik Regresyon (LR), veri seti içerisindeki tüm değişkenleri sayısal olarak kabul eden ve normal bir dağılıma sahip olan, ikili sınıflandırma algoritması olarak tanımlanmaktadır. Bu varsayıma rağmen, LR ile normal dağılım göstermeyen veriler üzerinde dahi iyi sonuçlar alınabilemektedir.

LR algoritması, doğrusal bir regresyon fonksiyonuna dayalıdır. Bu fonksiyon, bir lojistik fonksiyon kullanılarak dönüştürülmüş her giriş değeri için bir katsayı öğrenme sisteme sahiptir. Hızlı ve basit bir yapıya sahip olmasına rağmen, bazı problemler üzerinde son derece etkili sonuçlar vermektedir. Ancak LR, yalnızca ikili sınıflandırma modellerini desteklemektedir [?].

LR, düzeltilmiş olasılık oranları hakkında çıkarım sağlamak amacıyla geliştirilmiş standart bir öğrenme algoritmasıdır [?]. LR, veri setinin kalitesine bağlı olarak ayrıca bir modeldir. Modelin belirlenmesinde aşağıdaki öğeler dikkate alınır:

- Özellik değerleri: X_1, X_2, \dots, X_n ,
- Ağırlık değerleri: W_1, W_2, \dots, W_n ,
- Sapma değerleri: b_1, b_2, \dots, b_n ,
- Sınıflar: 1/0.

Model, aşağıdaki eşitlik ile ifade edilebilir:

$$P(Y = 1|X) = \frac{1}{1 + e^{-(W_0 + W_1 X_1 + W_2 X_2 + \dots + W_n X_n)}} \quad (3)$$

Bu denklemde, $P(Y = 1|X)$ gözlemin sınıfı ait olma olasılığını temsil eder.

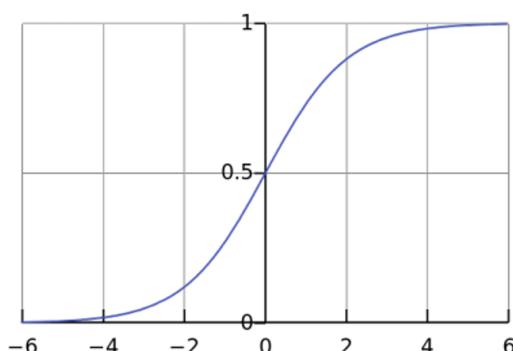


Fig. 18: Lojistik Regresyon Grafiği

LR, logaritmik ilerleme gerçekleştiren bir *logit* fonksiyonudur. Yapısı itibarıyle, Fig 18'teki grafikte gösterilen eğriye en iyi şekilde benzeyen değerleri tespit etmeye çalışır.

Özetle Logistic Regression (Lojistik Regresyon), bağımlı bir değişkenin kategorik olarak sınıflandırılmasını sağlayan bir istatistiksel yöntemdir. Çoğunlukla ikili sınıflandırma problemlerde kullanılmakla birlikte, birden fazla sınıfı ayırt etmek için genelleştirilmiş versiyonları da mevcuttur.

Enerji tüketim verilerinde lojistik regresyon, örneğin bir cihazın enerji verimli olup olmadığını tahmin edilmesinde veya belirli bir enerji tüketim seviyesine ait sınıflandırma yapılmasında etkili bir yöntemdir. Bu algoritma, sigmoid fonksiyonu kullanarak tahmin edilen değerleri 0 ile 1 arasında sıkıştırır ve bu sayede sonuçların olasılık şeklinde yorumlanması sağlar.

Lojistik regresyonun avantajlarından biri, hesaplama açısından kolay ve hızlı bir yöntem olmasıdır. Ayrıca, verilerin doğrusal olarak ayrılabilir olması durumunda oldukça başarılı sonuçlar verir. Bununla birlikte, doğrusal olmayan veriler üzerinde performansı sınırlı olabilir, bu nedenle polinomsal özellikler eklenerek veya daha karmaşık modellerle desteklenerek kullanılabilir.

3) Decision Tree: Karar Ağacı (KA), sınıflandırma ve regresyon problemlerinin çözümlerini destekleyebilmektedir. Bu yöntem, veri örneklerini değerlendirmek amacıyla bir ağaç yapısının oluşturulmasına dayanır. KA, ters çevrilmiş bir ağacın köküyle başlar ve bir tahmin sonucuna ulaşılana kadar dallanarak devam eder. Doğru tahminlere ulaşabilmek için, ağacın oluşturulması sırasında en iyi ayırcılık özelliğine sahip olan niteliğin tespit edilmesi sağlanır.

Dağılım tabanlı tahminlerde, giriş verilerinin tamamı üzerinde bir modelin geçerli olduğu varsayılar. Veri setine ait parametrelerin öğrenilebilmesi için bütün veri seti kullanılır. Öğrenme işlemi tamamlandıktan sonra, test verilerinde de aynı yapı ve öğrenilmiş parametreler kullanılarak sistem sinanır. Buna karşılık, dağılıma bağımlı olmayan tahmin süreçlerinde belirlenmiş bir ölçüt (örneğin, Öklid uzaklılığı) ile öğrenme seti yerel parçalara ayrılır. Giriş verisi, kendisine en yakın alanla eşleştirilir ve bu alanda eğitilmiş lokal bir model kullanılır. KA, bu yapısı sayesinde dağılımdan bağımsız çalışabilmektedir. Çünkü öğrenme sürecinin başlangıcında sınıf dağılımları ile ilgili herhangi bir tahminde bulunmaz. Ağacın yapısı, veri setinin özelliklerine göre dallar ve yapraklar eklenerek dinamik bir şekilde oluşturulur.

Karar ağacı oluşturulması için C4.5 ve ID3 gibi çeşitli algoritmalar kullanılmaktadır. Bu çalışma kapsamında C4.5 algoritması kullanılmıştır. C4.5 algoritması, giriş verilerinin sıklıklarına göre sınıflandırma işlemini gerçekleştiren ve entropi hesabına dayanan bir yöntemdir. Algoritmanın çalışma yapısı şu şekilde özetlenebilir:

- 1) Veri setindeki tüm nitelikler için (6) numaralı eşitlik kullanılarak entropi hesabı yapılır.
- 2) Her bir nitelik için hesaplanan entropi değeri, sınıfın entropi değerine bölünerek bilgi kazancı değerleri hesaplanır.
- 3) En yüksek bilgi kazancı değerine sahip nitelik kök olarak seçilir ve dallanma başlar.
- 4) Kök olarak seçilen niteliğin değerleri dışındaki diğer nitelikler için aynı işlemler tekrarlanır.
- 5) Tüm veriler işlenip yapraklara ulaşıldığında ağaç tamamlanmış olur.

Entropi, aşağıdaki formül ile hesaplanır:

$$\text{Entropi} = - \sum_{i=1}^n P_i \log_2 P_i \quad (4)$$

Burada P_i , i -inci sınıfın olasılığıdır.

Özetle Decision Tree (Karar Ağacı), veri setindeki örüntülerini belirlemek ve sınıflandırma veya regresyon problemlerini çözmek için kullanılan sezgisel bir algoritmadır. Karar ağacı, veri setini dallara ayırarak çalışır ve her bir dal bir karar noktasını veya bir sonucu temsil eder.

Enerji tüketim verilerinin analizi bağlamında, Decision Tree algoritması, bir cihazın belirli bir enerji tüketim seviyesine ait olup olmadığını tahmin etmek veya operasyonel süreçlerdeki anomalilikleri tespit etmek için kullanılabilir. Bu algoritma, verilerdeki karmaşıklıkları kolaylıkla yakalayabilir ve görselleştirilebilir bir yapı sunduğu için anlaşılması oldukça kolaydır.

Karar ağacının temel avantajlarından biri, kategorik ve sürekli verilerle çalışabilmesidir. Ayrıca, veri ön işleme gereksiniminin düşük olması ve özellikle önem sırasına göre değerlendirebilmesi, bu algoritmayı popüler kılar. Ancak, karar ağaçları aşırı öğrenmeye (overfitting) eğilimlidir; bu durum, ağaç derinliğinin kontrol edilmesi veya budama (pruning) tekniklerinin kullanılması ile önlenebilir.

4) Random Forest: Rastgele Orman (RO), birden fazla karar ağacının birleşiminden oluşan bir sınıflandırma algoritmasıdır. Orijinal veri setinden rastgele seçilmiş ve birbirinden bağımsız alt parçalar kullanılarak, ormanın içerisindeki ağaçların eğitilmesi esasına dayanır.

RO algoritmasında, ağaçların büyümesi sırasında rastgele özellik seçimi yapılır. Bunun sebebi, büyük veri setleri üzerinde tek bir karar ağacının yeterli doğruluk sağlayamaması riskidir. Bu durum, veri setinin alt parçalara ayrılp, her bir parçanın farklı ağaçlar tarafından öğrenilmesiyle doğruluk oranlarının artırılmasını sağlar.

Rastgele özellik seçim süreci ile ormanın içerisindeki ağaçlar, değiştirme yöntemi (bootstrap) kullanılarak eğitim özelliklerinin alt kümelerinden oluşturulur. Bu yöntem, aynı özelliğin birden fazla kez seçilebilmesine veya bazı özelliklerin hiç seçilmemesine neden olabilir. Bu durum, daha hızlı ve gürültüye dayanıklı öğrenme modelleri oluşturulmasını sağlar.

RO algoritmasının bir diğer avantajı, ormanın içerisindeki ağaçların modele esneklik kazandırmasıdır. Bu durum, sınıflandırma, kümeleme ve regresyon işlemlerinin performansını artırır. Özellikle büyük veri setlerinin değerlendirilmesi aşamasında RO, oldukça iyi bir seçim olarak değerlendirilmektedir.

RO, ağaç yapılı sınıflandırıcıların birleşiminden oluşur. Bu ağaç yapılı sınıflandırıcılar, birbirlerinden bağımsızdır ancak aynı şekilde dağıtılmış rastgele vektörleri içerir. Her ağaç, giriş verilerindeki en popüler sınıf değeri için oy verme süreci uygular. Ağaçlardan alınan oyalar sonucunda, en fazla oyu alan sınıf etiketi, RO algoritması tarafından belirlenmiş etiket değeri olarak gösterilir.

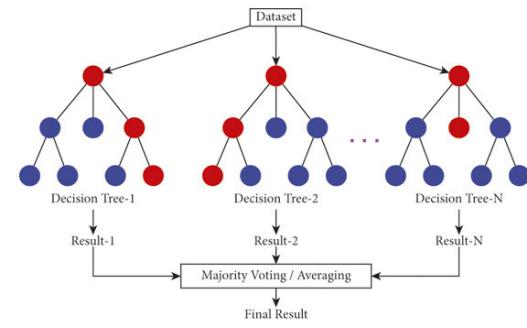


Fig. 19: Rastgele Orman yapısının bir örneği.

Özetle Random Forest (Rastgele Orman), birden fazla karar ağacının birleşiminden oluşan ve ensemble learning (topluluk öğrenimi) yöntemini kullanan güçlü bir sınıflandırma ve regresyon algoritmasıdır. Rastgele orman, farklı veri alt kümeleri ve özellikler üzerinde eğitilen çok sayıda karar ağacını bir araya getirerek nihai tahmini gerçekleştirir.

Enerji tüketim verilerinde Random Forest algoritması, örneğin enerji verimliliği sınıflandırması veya enerji tüketim seviyelerinin tahmini gibi uygulamalarda kullanılabilir. Algoritmanın temel çalışma prensibi, her bir karar ağacının bağımsız olarak eğitim aldığı ve sonuçların birleştirildiği bir yapıya dayanır. Bu birleştirme işlemi, sınıflandırma problemlerinde çoğunluk oylaması, regresyon problemlerinde ise ortalama alma yöntemiyle gerçekleştirilir.

Random Forest'in temel avantajları arasında yüksek doğruluk oranı, overfitting'e karşı dayanıklılık ve çok boyutlu verilerle etkili bir şekilde çalışabilmesi yer alır. Ayrıca, model, özellik önem derecelerini belirleyebilmesi sayesinde açıklanabilirlik sağlar. Bununla birlikte, çok sayıda ağaç kullanılması, algoritmanın hesaplama maliyetini artırabilir; bu durum büyük veri setlerinde performans iyileştirme teknikleri ile dengelenebilir.

5) K-Nearest Neighbors (KNN): k-NN algoritması, sınıflandırma teknikleri arasında benzerlik fonksiyonlarını çalıştırarak tahmin süreçlerini gerçekleştiren bir algoritmadır. İlk olarak 1950'li yıllarda keşfedilmiş ve günümüzde hala yaygın olarak kullanılmaktadır. Algoritmanın çalışma mantığı, iki boyutlu bir düzleme üzerinden anlaşılabılır. Şekil 9'da görüldüğü üzere, veriler iki boyutlu bir düzleme yerleştirilmekte ve dikey (x) ile yatay (y) eksen değerlerine göre benzerlik hesaplamaları yapılmaktadır.

k-NN algoritması, makine öğrenmesi algoritmaları arasında en basit ve etkili sınıflandırıcı türlerinden biridir. Çalışma prensibi, bir nesnenin sınıflandırılması sırasında k adet komşunun oylarına dayanmaktadır. k , genellikle küçük bir pozitif tam sayıdır ve sınıf tahmininde çoğunluğun kararını temsil eder.

Denetimli bir algoritma olarak k-NN, sınıf etiketi bilinmeyen bir örnek veri için, bu verinin en yakın komşuları tarafından sınıflandırılmasını sağlar. Algoritma, test verileriyle sınıandığında ya da yeni bir verinin sınıf etiketi belirlenmek istediğiinde, bu verilerin öğrenme kümelerindeki verilerle olan

mesafelerini ölçer. En yakın k komşu, sınıflandırma veya tahmin işlemleri için kullanılır. Mesafe hesaplamasında genellikle Öklid uzaklığı eşitliği tercih edilmektedir:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (5)$$

k -NN algoritmasının temel varsayımları, verilerin belirli bir özellik alanına ait olduğunu söylemektedir. Bu nedenle, yeni bir veri noktası için, öğrenme kümesindeki tüm noktalara olan uzaklıklar tek tek hesaplanır. Eğer $k = 1$ olarak seçilmişse, yalnızca en yakın komşunun sınıf etiketi tahmin için kullanılır. Ancak k değerinin dikkatli bir şekilde seçilmesi önemlidir. Çok küçük bir k değeri, algoritmanın gürültüye duyarlı hale gelmesine neden olur. Çok büyük bir k değeri ise uzak komşuların karar sürecine dahil olması nedeniyle hata oranını artırabilir.

k -NN algoritmasının çalışma süreci şu adımlardan oluşur:

- 1) k değeri belirlenir.
- 2) Giriş verisi ile öğrenme kümesi verileri arasındaki mesafeler hesaplanır.
- 3) Hesaplanan mesafeler sıralanır.
- 4) En yakın k komşu belirlenir.
- 5) Fig 20'de gösterildiği gibi, basit çoğunluk yöntemi ile komşular arasında en fazla bulunan sınıf değeri, giriş verisi için tahmin edilir.

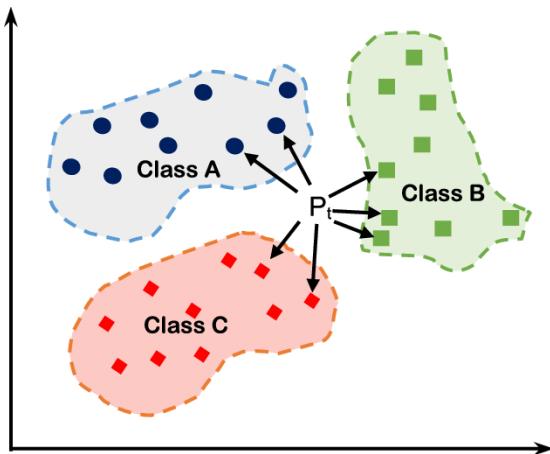


Fig. 20: k -NN Algoritması için Örnek Şema.

Özetle K-Nearest Neighbors (KNN), örneklerin sınıflandırılmasında veya regresyon problemlerinde kullanılan basit ama etkili bir makine öğrenimi algoritmasıdır. KNN, veri noktalarının sınıfını belirlemek için komşuluk ilişkilerini kullanır ve genellikle çok boyutlu veri kümelerinde tercih edilir.

Enerji tüketim verilerinde KNN algoritması, örneğin bir cihazın enerji verimli olup olmadığını veya belirli bir enerji tüketim kategorisine ait olup olmadığını tahmin etmek için kullanılabilir. Algoritmanın çalışma prensibi, bir veri noktasına en yakın k komşunun sınıfına bakarak tahmin yapmasıdır. Sınıflandırma problemlerinde çoğunluk oylaması, regresyon problemlerinde ise komşu değerlerin ortalaması alınır.

KNN'nin temel avantajları, parametresiz bir yöntem olması ve herhangi bir model eğitimi gerektirmemesidir. Ancak, büyük veri kümelerinde ve yüksek boyutlu verilerle çalışırken hesaplama maliyeti artabilir. Bunun yanı sıra, uygun k değerinin seçimi, algoritmanın performansı üzerinde önemli bir etkiye sahiptir. Genelde k değerinin optimizasyonu için çapraz doğrulama yöntemi kullanılır.

6) **Support Vector Machines (SVM):** Destek Vektör Makineleri (DVM), LR algoritması gibi ayrımcı bir model olarak çalışmaktadır. Bu algoritma, regresyon, aykırı veri tespiti ve sınıflandırma işlemleri için denetimli bir öğrenme sistemi sunmaktadır.

DVM, 1995 yılında Vladimir Vapnik tarafından tanıtılmıştır ve sınıflandırma ile örüntü tanıma süreçlerinde kullanılan basit ve verimli bir algoritmadır. DVM'nin temel amacı, hiper düzlemleri ve sınırları ortaya çıkaracak fonksiyonların elde edilmesidir. Bu hiper düzlemler, istatistiksel öğrenme yöntemleri ile eğitilir ve giriş veri noktalarının farklı kategorilere ayırtılmasını sağlar.

Destek Vektör Makineleri, prensip olarak istatistiksel öğrenme yöntemlerine ve yapısal risklerin minimize edilmesine dayanır. Büyük veri setlerini kullanarak hızlı öğrenme kapasitesine sahip olması, algoritmanın temel avantajlarından biridir. Bu özellik, doğrusal olmayan yüksek boyutlu veri modelleme problemlerine çözüm getirme imkanı sağlar.

DVM, genellikle sınıflandırma problemlerinde kullanılan bir algoritmadır. Çalışma sisteminde, n bağımsız değişken sayısı olmak üzere tüm veri noktalarının n -boyutlu uzaydaki koordinatları, değişken değerleri olarak tanımlanır. Daha sonra, iki sınıfı birbirinden ayıran iki boyutlu bir hiper düzlemler belirlenir. Sınıflandırma işlemi, Fig 21'de gösterildiği gibi doğrusal hiper düzlemler kullanılarak gerçekleştirilebilir.

Ancak, tüm verilerin doğrusal hiper düzlemler ile sınıflandırılması her zaman mümkün değildir. Doğrusal olmayan veri türleri için kernel fonksiyonu uygulanır. Kernel fonksiyonu, yüksek boyutlu uzayda doğrusal olmayan bir hiper düzlemler oluşturarak, bu tür verilerin sınıflandırılmasını sağlar. Bu yaklaşım, doğrusal olmayan bir problemin kernel hilesi aracılığıyla çözülmeyi mümkün kılar. Fig 22, doğrusal olmayan hiper düzlemler kullanılarak yapılan sınıflandırmayı görselleştirmektedir.

Özetle Support Vector Machines (SVM), sınıflandırma ve regresyon problemlerinde kullanılan güçlü bir makine öğrenimi algoritmasıdır. SVM, veri noktalarını sınıflar arasında en iyi ayırmayı sağlayacak şekilde bir hiper düzlemlerle ayırmaya çalışır.

Enerji tüketim verilerinde SVM, örneğin bir cihazın enerji verimli olup olmadığını veya anormal enerji tüketim davranışlarını tespit etmek için kullanılabilir. Algoritma, doğrusal olmayan veri kümelerinde de etkili olabilmesi için çekirdek (kernel) fonksiyonlarını kullanır. Bu fonksiyonlar, verileri daha yüksek boyutlu bir uzaya taşıyarak doğrusal ayırmayı sağlayabilir.

SVM'nin avantajları arasında yüksek doğruluk oranı, genelleme yeteneği ve karmaşık verilerle başa çıkabilme kabiliyeti bulunur. Ancak, büyük veri setlerinde hesaplama

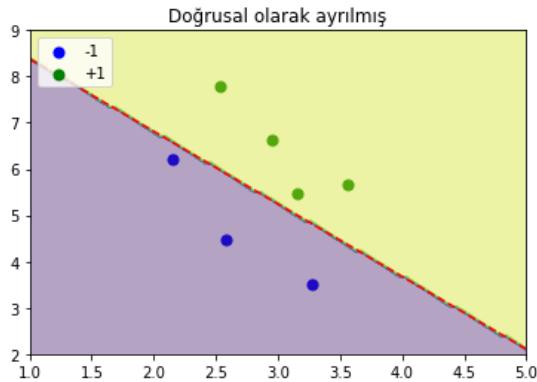


Fig. 21: Doğrusal Hiper Düzlem ile Sınıflandırma.

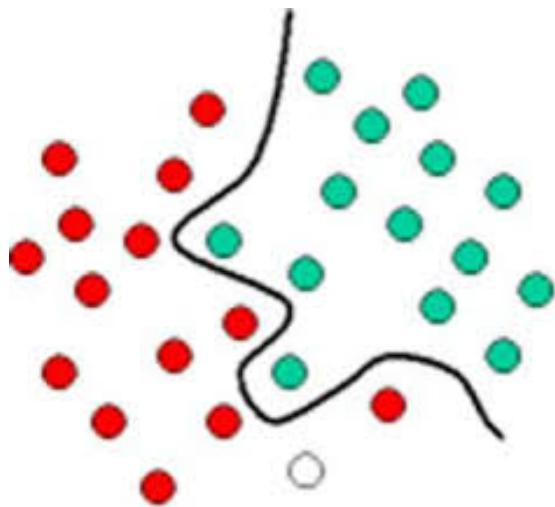


Fig. 22: Doğrusal Olmayan Hiper Düzlem ile Sınıflandırma.

maliyetleri artabilir ve uygun parametrelerin seçimi (örneğin, C ve gamma değerleri) algoritmanın performansı üzerinde önemli bir etkiye sahiptir. Bu parametreler genellikle çapraz doğrulama ile optimize edilir.

7) **Autoencoder:** Autoencoder, denetimsiz öğrenme algoritmalarından biri olup, giriş verisini yeniden oluşturmayı amaçlayan bir yapay sinir ağı modelidir. Autoencoder’lar, bir veri setindeki temel özelliklerini öğrenerek veriyi sıkıştırarak ve yeniden yapılandırmak için kullanılır. Bu algoritma, özellikle boyut indirgeme, gürültü temizleme ve veri görselleştirme gibi uygulamalarda yaygın olarak kullanılmaktadır.

Autoencoder modeli, genellikle üç temel bileşenden oluşur:

- 1) **Kodlayıcı (Encoder):** Giriş verisini daha düşük boyutlu bir temsil (latent uzay) haline dönüştürür.
- 2) **Kod (Latent Representation):** Verinin sıkıştırılmış halı olup, giriş verisinin temel özelliklerini içerir.
- 3) **Kod Çözücü (Decoder):** Koddan yola çıkarak giriş verisini yeniden oluşturur.

Modelin çalışma prensibi, giriş verisi x ’i bir kodlayıcı fonksiyonu f ile kodlamak ve ardından bir kod çözücü

fonksiyonu g ile yeniden oluşturmak şeklinde özetlenebilir:

$$h = f(x), \quad \hat{x} = g(h) \quad (6)$$

Burada h , giriş verisinin kodlanmış halini ifade ederken, \hat{x} , giriş verisinin yeniden oluşturulmuş halidir. Amaç, giriş verisi x ile yeniden oluşturulan veri \hat{x} arasındaki farkı minimize etmektir. Bu fark genellikle aşağıdaki kayıp fonksiyonu ile hesaplanır:

$$L(x, \hat{x}) = \|x - \hat{x}\|^2 \quad (7)$$

Bu denklemde L , giriş ve yeniden oluşturulan veri arasındaki kare hata (Mean Squared Error) kaybını ifade eder.

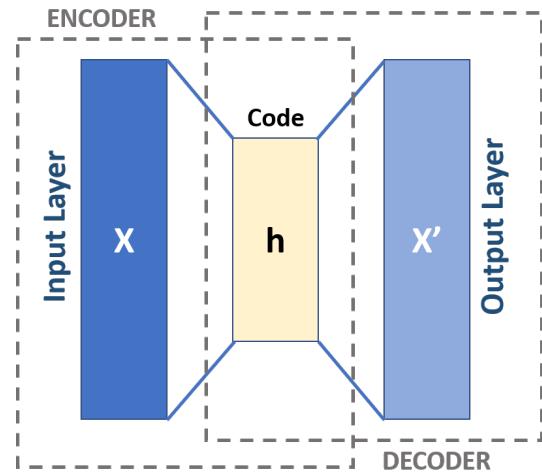


Fig. 23: Autoencoder yapısının genel bir şeması.

Özette Autoencoder, denetimsiz öğrenme (unsupervised learning) yöntemlerinden biri olup, verilerin sıkıştırılması ve yeniden yapılandırılması üzerine çalışan bir yapay sinir ağıdır. Genellikle veri boyutunu düşürmek, gürültü gidermek veya özellik çıkarımı yapmak için kullanılır.

Enerji tüketim verilerinde Autoencoder, anormal tüketim davranışlarını tespit etmek, enerji verilerinin sıkıştırılmış bir temsiline ulaşmak veya özellik mühendisliği amacıyla kullanılabilir. Model, bir encoder (kodlayıcı) ve decoder (çözücü) olmak üzere iki bileşenden oluşur. Encoder, giriş verisini sıkıştırarak gizli bir temsile dönüştürürken; decoder, bu sıkıştırılmış temsili kullanarak orijinal veriyi yeniden oluşturur.

Autoencoder’ın avantajları arasında, verilerin düşük boyutlu temsillerine ulaşmada etkin olması ve gürültülü veri kümeleri üzerinde çalışabilmesi yer alır. Bununla birlikte, modelin başarılı olması için uygun bir mimari ve hiperparametre ayarları gereklidir. Ayrıca, anomalî tespiti için kullanılan Autoencoder, yeniden yapılandırma hatasını analiz ederek anomalileri tespit eder.

8) **Convolutional Neural Networks (CNN):** Convolutional Neural Networks (CNN), görüntü tanıma ve işleme gibi yapılandırılmış veri üzerinde çalışan ve derin öğrenme yöntemlerinden biri olan bir yapay sinir ağı modelidir. CNN’ler, özellikle görüntü sınıflandırma, nesne algılama ve

segmentasyon gibi alanlarda yaygın olarak kullanılmaktadır. Bu ağlar, giriş verisindeki uzaysal ve zamansal özellikleri algılamak için konvolüsyon katmanları kullanır.

CNN, genellikle üç temel katmandan oluşur:

- Konvolüsyon Katmanı (Convolutional Layer):** Giriş verisi üzerinde filtreler (kernel) uygulanarak özellik haritaları (feature map) oluşturulur. Bu katmanda, aşağıdaki denklem kullanılarak bir giriş verisi I üzerinde birfiltre K ile konvolüsyon işlemi gerçekleştirilir:

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(i+m, j+n)K(m, n) \quad (8)$$

- Havuzlama Katmanı (Pooling Layer):** Özellik haritalarını küçüterek boyut azaltımı yapar ve modelin hesaplama maliyetini düşürür. En yaygın kullanılan yöntemlerden biri maksimum havuzlama (max pooling) yöntemidir.

- Tam Bağlantılı Katman (Fully Connected Layer):** Bu katman, ağıın çıktısını sınıflandırma veya regresyon problemlerinde kullanabilecek bir formata dönüştürür.

CNN'de doğrusal olmayan özellikleri öğrenmek için genellikle aşağıdaki aktivasyon fonksiyonları kullanılır:

- **ReLU (Rectified Linear Unit):**

$$f(x) = \max(0, x) \quad (9)$$

- **Sigmoid:**

$$f(x) = \frac{1}{1 + e^{-x}} \quad (10)$$

- **Softmax:** Çıkış katmanında sınıf olasılıklarını normalleştirerek için kullanılır.

CNN'ler aşağıdaki avantajları sunar:

- **Parametre Paylaşımı:** Filtrelerin yeniden kullanılması, parametrelerin ve hesaplama maliyetinin azaltılmasını sağlar.
- **Yerel Bağımlılık:** Görülerdeki piksel ilişkilerini algılayarak daha anlamlı özellikler çıkarır.
- **Ölçeklenebilirlik:** Farklı boyutlardaki veri setleriyle çalışabilir.

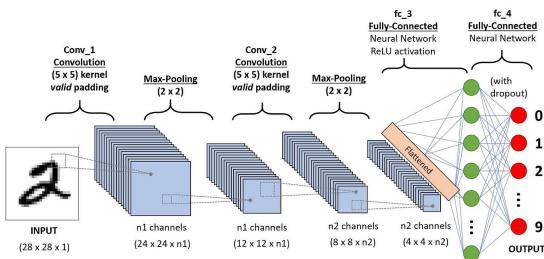


Fig. 24: Convolutional Neural Networks (CNN) yapısının genel bir şeması.

Convolutional Neural Networks, modern derin öğrenme uygulamalarında güçlü ve esnek bir yöntem sunmaktadır.

Özellikle büyük veri setleri üzerinde çalışırken sağladığı performans, CNN'lerin çeşitli problemler için tercih edilmesine neden olmaktadır.

Özette Convolutional Neural Networks (CNN), genellikle görüntü verilerinin işlenmesi için kullanılan derin öğrenme modellerinden biridir. Ancak CNN, enerji tüketim verilerinin zaman serileri gibi birden fazla boyut içeren yapılarında da kullanılabilir. CNN'nin temel yapı taşı olan konvolüsyon katmanları, yerel özelliklerin çıkarılmasını ve veri boyutunun azaltılmasını sağlar.

Enerji tüketim verilerinde CNN, örneğin enerji kullanımındaki örüntülerini tespit etmek veya anomalileri belirlemek için kullanılabilir. Modelin özellik öğrenme kabiliyeti, karmaşık veri yapılarının analizinde avantaj sağlar.

CNN'nin temel avantajları arasında, büyük veri kümeleri ile çalışabilmesi ve karmaşık örüntülerin yakalayabilmesi bulunur. Bununla birlikte, modelin eğitim süreci yoğun hesaplama gerektirir ve hiperparametre ayarlarının dikkatle yapılması gereklidir. Ayrıca, CNN'nin performansı, veri ön işleme ve mimari seçimine bağlı olarak büyük ölçüde değişebilir.

9) **Recurrent Neural Networks (RNN):** Recurrent Neural Networks (RNN), özellikle sıralı veriler üzerinde çalışan ve her bir zaman adımında önceki zaman adımından gelen bilgiyi dikkate alarak öğrenme yapan bir yapay sinir ağı modelidir. RNN, doğal dil işleme (NLP), zaman serisi analizi, konuşma tanıma gibi sıralı veri gerektiren problemlerde yaygın olarak kullanılmaktadır.

RNN'nin temel farkı, döngüsel bir yapıya sahip olmasıdır. Bu yapı, geçmiş bilgiye dayalı öğrenme yapılmasını sağlar. Her bir zaman adımında, ağıın durumu şu şekilde güncellenir:

$$h_t = f(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (11)$$

Burada:

- h_t : t -inci zaman adımındaki gizli durum (hidden state),
- x_t : t -inci zaman adımındaki giriş,
- W_{xh} : Giriş ile gizli durum arasındaki ağırlık matrisi,
- W_{hh} : Gizli durumun kendi ağırlık matrisi,
- b_h : Bias terimi,
- f : Aktivasyon fonksiyonu (genellikle tanh veya ReLU kullanılır).

Modelin çıktısı ise:

$$y_t = g(W_{hy}h_t + b_y) \quad (12)$$

şeklinde hesaplanır. Burada W_{hy} , gizli durum ile çıktı arasındaki ağırlık matrisidir ve g , genellikle softmax fonksiyon gibi doğrusal olmayan bir dönüşüm fonksiyonudur.

RNN'nin temel avantajı, sıralı verilerdeki bağımlılıkları öğrenebilmesidir. Ancak, uzun dizilerde **kaybolan gradyan (vanishing gradient)** ve **patlayan gradyan (exploding gradient)** problemleri yaşanabilir. Bu sorunları gidermek için Long Short-Term Memory (LSTM) ve Gated Recurrent Unit (GRU) gibi RNN türleri geliştirilmiştir.

RNN'nin farklı türleri aşağıda sıralanmıştır:

- **Many-to-One:** Birden fazla zaman adımından alınan veriler bir tek çıktı üretir. Örneğin, duyu analizi.

- **One-to-Many:** Tek bir girişten birden fazla çıktı üretilir. Örneğin, metin oluşturma.
- **Many-to-Many:** Hem giriş hem de çıkış birden fazla zaman adımını kapsar. Örneğin, makine çevirisii.

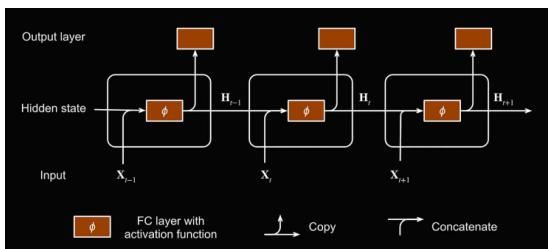


Fig. 25: Recurrent Neural Network (RNN) yapısının genel bir şeması.

Recurrent Neural Networks, sıralı veri işlemede güçlü bir araçtır. Ancak, temel RNN'nin uzun dizilerdeki öğrenme sorunları nedeniyle LSTM ve GRU gibi türevlerinin kullanımı daha yaygındır. RNN'nin esnek yapısı, birçok uygulamada etkili sonuçlar elde edilmesini sağlamaktadır.

Özette Recurrent Neural Networks (RNN), zaman serisi verilerini işlemek için tasarlanmış bir yapay sinir ağı türüdür. RNN, önceki zaman adımlarından gelen bilgileri kullanarak gelecekteki tahminler yapabilir ve bu özellikleyle enerji tüketim verilerindeki örüntülerin analizi için uygundır.

Enerji tüketim verilerinde RNN, özellikle zaman serisi tahmini, anomalî tespiti ve enerji kullanımında kısa vadeli öngörülerde kullanılabilir. Model, sıralı verilerle çalışmak için geri bildirim döngüleri içerir ve bu döngüler sayesinde geçmiş bilgiler ağda tutulabilir.

RNN'nin avantajları arasında, sıralı ve zaman serisi verilerinde etkili olması ve veri bağımlılıklarını öğrenme kabiliyeti yer alır. Bununla birlikte, uzun süreli bağımlılıkları öğrenmede zorluk çeker ve vanishing gradient (kaybolan gradyan) problemi yaşayabilir. Bu sorunları çözmek için Gated Recurrent Unit (GRU) ve Long Short-Term Memory (LSTM) gibi RNN türevleri geliştirilmiştir.

10) **Gated Recurrent Unit (GRU):** Gated Recurrent Unit (GRU), Recurrent Neural Networks (RNN) ailesine ait bir sınırlı ağı modelidir. GRU, uzun süreli bağımlılıkları öğrenmek ve RNN'lerdeki kaybolan gradyan problemine çözüm getirmek amacıyla geliştirilmiştir. 2014 yılında Cho ve arkadaşları tarafından önerilen GRU, Long Short-Term Memory (LSTM) ile benzer bir yapıya sahiptir ancak daha az karmaşık ve daha az parametre ile çalışır.

GRU, LSTM'den farklı olarak ayrı bir hücre durumu (\$c_t\$) yerine yalnızca gizli durum (\$h_t\$) kullanır ve iki temel kapiya (gate) sahiptir:

- **Güncelleme Kapısı (Update Gate, \$z_t\$):** Hangi bilginin bir sonraki duruma aktarılacağını belirler.
- **Sıfırlama Kapısı (Reset Gate, \$r_t\$):** Geçmiş durumun ne kadarının unutulacağını kontrol eder.

Bu kapılar ve GRU'nun çalışma mekanizması şu şekilde ifade edilebilir:

1) Güncelleme Kapısı:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \quad (13)$$

2) Sıfırlama Kapısı:

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \quad (14)$$

3) Aday Gizli Durum (Candidate Hidden State):

$$\tilde{h}_t = \tanh(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h) \quad (15)$$

4) Gizli Durumun Güncellenmesi:

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t \quad (16)$$

Burada:

- \$x_t\$: \$t\$-inci zaman adımındaki giriş,
- \$h_t\$: \$t\$-inci zaman adımındaki gizli durum,
- \$W, U, b\$: Ağırlık ve bias matrisleri,
- \$\sigma\$: Sigmoid aktivasyon fonksiyonu,
- \$\tanh\$: Hiperbolik tanjant fonksiyonu,
- \$\odot\$: Element bazında çarpma işlemidir.

GRU, aşağıdaki avantajları ve farkları ile öne çıkar:

- **Daha Az Karmaşıklık:** LSTM'den daha az parametreye sahiptir, bu nedenle hesaplama maliyeti daha düşüktür.
- **Uzun Bağımlılıkları Öğrenme:** Kapılar, önemli bilgilerin uzun süre tutulmasını sağlar.
- **Dinamik Bellek:** Sıfırlama ve güncelleme kapıları, ağır hem kısa hem de uzun süreli bağımlılıkları öğrenmesini sağlar.

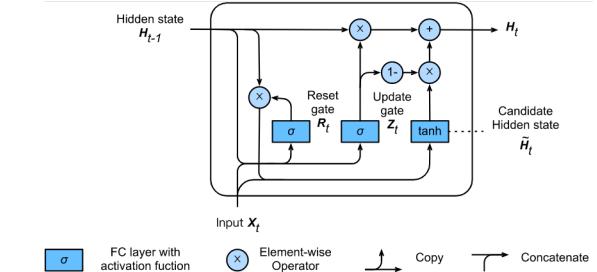


Fig. 26: Gated Recurrent Unit (GRU) yapısının genel bir şeması.

Gated Recurrent Unit (GRU), RNN'lerin daha verimli bir sürümü olarak birçok sıralı veri problemine çözüm sunmaktadır. Daha az parametre ve daha az hesaplama maliyeti ile LSTM'ye kıyasla pratik avantajlar sağlamaktadır.

Özette Gated Recurrent Unit (GRU), Recurrent Neural Networks (RNN) mimarisinin bir diğer geliştirilmiş versiyonudur ve LSTM'ye benzer şekilde uzun vadeli bağımlılıkları öğrenmek için tasarlanmıştır. GRU, enerji tüketim verilerindeki zaman serisi tahmini ve anomalî tespiti gibi uygulamalarda yaygın olarak kullanılmaktadır.

GRU, LSTM'ye kıyasla daha basit bir yapıya sahiptir. LSTM'deki giriş, unutma ve çıkış kapıları yerine, GRU sadece güncelleme ve sıfırlama kapılarına sahiptir. Bu daha basit yapı, modelin daha hızlı eğitim olmasını ve daha az hesaplama kaynağı gerektirmesini sağlar.

Enerji verilerindeki uygulamalarda, GRU, zaman serisi analizi ve enerji tüketim öngörülerinde etkili bir yöntemdir. Özellikle, verilerin daha az karmaşık olduğu durumlarda GRU, LSTM'ye kıyasla benzer performans sağlayabilir.

GRU'nun avantajları arasında, basit yapısı sayesinde hızlı eğitim süreci, uzun vadeli bağımlılıkları öğrenme kabiliyeti ve daha az hesaplama gereksinimi yer alır. Ancak, modelin performansı, veri setinin karmaşıklığına ve model mimarisine bağlı olarak değişebilir.

11) **Vanilla Transformer:** Vanilla Transformer, doğal dil işleme (NLP) ve diğer sıralı veri problemleri için kullanılan, dikkat mekanizmasına (attention mechanism) dayalı bir derin öğrenme modelidir. İlk olarak 2017 yılında Vaswani ve arkadaşları tarafından "Attention is All You Need" başlıklı çalışmada tanıtılmıştır. Transformer modeli, RNN veya CNN gibi sıralı işleme yöntemlerine ihtiyaç duymadan, paralel hesaplama yeteneğiyle verimli bir şekilde çalışır.

Vanilla Transformer modeli, encoder ve decoder olmak üzere iki temel bileşenden oluşur:

- 1) **Encoder:** Giriş dizisini özellik vektörlerine dönüştürür. Birden fazla encoder bloğundan oluşur ve her bir blok iki alt katman içerir:
 - Çoklu-başlı Dikkat Mekanizması (Multi-Head Attention)
 - Konumlu İleri Beslemeli Ağ (Position-wise Feed-Forward Network)

- 2) **Decoder:** Encoder'dan gelen özellik vektörlerini kullanarak hedef diziyi tahmin eder. Decoder bloğu da iki alt katmandan oluşur:

- Maskelenmiş Çoklu-başlı Dikkat Mekanizması
- Encoder-Decoder Dikkat Mekanizması
- Konumlu İleri Beslemeli Ağ

Transformer modelinin temelinde dikkat mekanizması yer almaktadır. Dikkat mekanizması, her bir kelimenin diğer kelimelerle olan ilişkisini öğrenir ve giriş verisindeki önemli unsurları öne çıkarır. Mekanizma şu şekilde ifade edilir:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^\top}{\sqrt{d_k}} \right) V \quad (17)$$

Burada:

- Q : Sorgu matrisi (Query),
- K : Anahtar matrisi (Key),
- V : Değer matrisi (Value),
- d_k : Anahtarların boyutudur.

Transformer modeli sıralı veriyi işlerken konum bilgisi kullanmaz. Bu nedenle, pozisyon bilgisi eklemek için pozisyon kodlaması (positional encoding) uygulanır. Pozisyon kodlaması, her bir kelimeye konum bilgisi eklemek için aşağıdaki gibi trigonometrik fonksiyonlar kullanır:

$$PE(pos, 2i) = \sin \left(\frac{pos}{10000^{2i/d}} \right), \quad (18)$$

$$PE(pos, 2i + 1) = \cos \left(\frac{pos}{10000^{2i/d}} \right) \quad (19)$$

Burada pos , kelimenin pozisyonunu; i , boyutunu; d , modelin boyutunu ifade eder.

Vanilla Transformer'ın avantajları şunlardır:

- **Paralel Hesaplama:** Tüm diziyi aynı anda işleyerek eğitimi hızlandırır.
- **Uzun Bağımlılıkları Öğrenme:** Dikkat mekanizması, uzak konumlar arasındaki ilişkileri etkili bir şekilde öğrenir.
- **Esneklik:** NLP, bilgisayarlı görüs (computer vision) gibi birçok farklı alanda uygulanabilir.

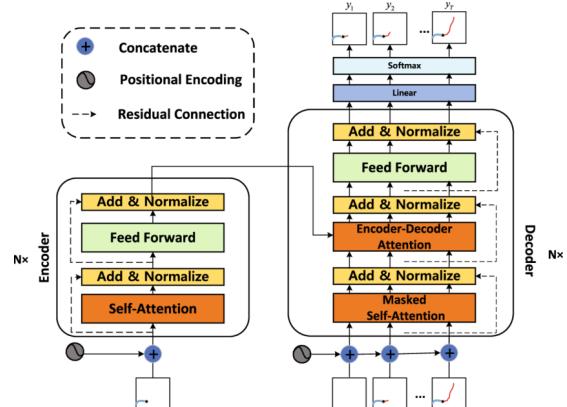


Fig. 27: Vanilla Transformer yapısının genel bir şeması.

Vanilla Transformer, dikkat mekanizmasını kullanarak RNN'lerin sıralı işleme bağımlılığını ortadan kaldırmış ve derin öğrenme modellerinde çığır açmıştır. Bu modelin temel yapısı, günümüzde birçok gelişmiş modelin (BERT, GPT) temelini oluşturmuştur.

Vanilla Transformer, doğal dil işleme (NLP) gibi sıralı veriler üzerinde kullanılan bir derin öğrenme modelidir. Transformer modeli, Recurrent Neural Networks (RNN) ve türevlerinden farklı olarak, sıralı verileri işlemek için tamamen dikkat (attention) mekanizmasına dayanır. Bu yaklaşım, paralel işlemeyi desteklediği için eğitim ve çıkarım sürecinde büyük bir hız avantajı sağlar.

Enerji tüketim verilerinde Vanilla Transformer, zaman serisi analizi, örtülü tanıma ve anomalî tespiti gibi uygulamalarda kullanılabilir. Transformer modelinin temel yapı taşları arasında çok başlı dikkat (multi-head attention), konum kodlamaları (positional encoding) ve tam bağlı katmanlar (fully connected layers) bulunur. Bu yapılar, verilerdeki uzun vadeli bağımlılıkları etkili bir şekilde modellemeye olanak tanır.

Vanilla Transformer'ın avantajları arasında, paralel işleme yeteneği, uzun vadeli bağımlılıkları öğrenme kapasitesi ve büyük ölçekli veri setleriyle etkin çalışabilme özelliği yer almaktadır. Bununla birlikte, Transformer modelleri, büyük miktarda veriye ve hesaplama kaynağına ihtiyaç duyar, bu da küçük veri setleri için kullanımını sınırlayabilir. Ayrıca, hiperparametre ayarları ve modelin ölçeklendirilmesi dikkatle alınmalıdır.

12) **Encoder-Decoder Transformer:** Encoder-Decoder Transformer, sıralı veri işlemede kullanılan, dikkat mekanizması (attention mechanism) temelli bir derin öğrenme modelidir. İlk kez 2017 yılında Vaswani ve arkadaşları

tarafından tanıtılan bu model, özellikle makine çevirisi, metin özetleme ve dil modelleme gibi doğal dil işleme (NLP) görevlerinde yüksek performans sağlamaktadır.

Encoder-Decoder Transformer modeli, iki temel bileşenden oluşur:

- 1) **Encoder:** Giriş dizisini alır, işlemlerden geçirir ve sabit boyutlu bir gizli temsil oluşturur. Birden fazla encoder bloğundan oluşur. Her encoder bloğu şu iki alt katmandan oluşur:
 - Çoklu-başlı Dikkat Mekanizması (Multi-Head Attention)
 - Konumlu İleri Beslemeli Ağ (Position-wise Feed Forward Network)
- 2) **Decoder:** Encoder'dan gelen gizli temsili alır ve hedef diziyi oluşturur. Decoder bloğu şu üç alt katmandan oluşur:
 - Maskelenmiş Çoklu-başlı Dikkat Mekanizması
 - Encoder-Decoder Dikkat Mekanizması
 - Konumlu İleri Beslemeli Ağ

Dikkat mekanizması, Transformer'in temel bileşenlerinden biridir. Encoder ve decoder içindeki dikkat mekanizması, dizinin her elemanını diğer elemanlarla ilişkilendirek bağlam bilgisi öğrenir. Mekanizma şu şekilde tanımlanır:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^\top}{\sqrt{d_k}} \right) V \quad (20)$$

Burada:

- Q : Sorgu matrisi (Query),
- K : Anahtar matrisi (Key),
- V : Değer matrisi (Value),
- d_k : Anahtarların boyutudur.

Transformer modeli, sıralı verinin düzenini korumak için pozisyon kodlaması kullanır. Bu kodlama, giriş verisine konum bilgisi eklemek için trigonometrik fonksiyonlar kullanır:

$$\begin{aligned} PE(pos, 2i) &= \sin \left(\frac{pos}{10000^{2i/d}} \right), \\ PE(pos, 2i + 1) &= \cos \left(\frac{pos}{10000^{2i/d}} \right) \end{aligned} \quad (21)$$

Burada pos , pozisyonu; i , boyut indeksini; d , modelin boyutunu ifade eder.

Encoder-decoder Transformer modeli, şu şekilde çalışır:

- 1) Giriş dizisi, embedding katmanı ve pozisyon kodlaması ile işlenerek encoder'a verilir.
- 2) Encoder, giriş dizisini işleyerek sabit boyutlu bir gizli temsili oluşturur.
- 3) Decoder, encoder'dan gelen temsil ile hedef diziyi maskelenmiş dikkat mekanizması kullanarak sırayla oluşturur.
- 4) Son çıktılar, softmax aktivasyon fonksiyonu ile olasılık dağılımına dönüştürülerek hedef dizi tahmin edilir.

Encoder-Decoder Transformer'ın avantajları şunlardır:

- **Paralel Hesaplama:** Encoder ve decoder yapılarını paralel olarak işletecek eğitim süresini kısaltır.

- **Uzun Bağımlılıkları Öğrenme:** Dikkat mekanizması sayesinde uzun sıralı bağımlılıkları etkili şekilde öğrenir.
- **Esneklik:** Farklı dil modelleme ve sıralı veri problemleri için uyarlanabilir.

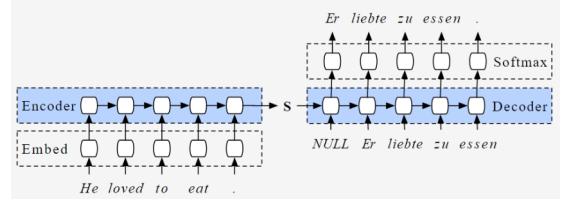


Fig. 28: Encoder-Decoder Transformer yapısının genel bir şeması.

Encoder-Decoder Transformer, modern derin öğrenme modellerinin temelini oluşturan, dikkat mekanizmasını etkin şekilde kullanan bir modeldir. Paralel hesaplama avantajı ve yüksek doğruluğu sayesinde NLP ve diğer sıralı veri problemlerinde yaygın olarak kullanılmaktadır.

Özette Encoder-Decoder Transformer, yeniden yapılandırma ve tahmin problemleri için tasarlanmıştır. Özellikle zaman serisi modellemelerinde anomali tespiti gibi alanlarda etkilidir. Bu model, encoder ve decoder bloklarından oluşur. Encoder, giriş verisini özetlerken, decoder bu özet üzerinden çıkış verisini oluşturur.

Zaman serisi verileri için orta düzeyde bir uygunluk sunar ve daha karmaşık yapıların temelini oluşturur. Tek veya çoklu giriş değişkenleri ile çalışabilir, bu da modelin esnekliğini artırır. Açıklanabilirlik seviyesi orta düzeydedir. Performans olarak Vanilla Transformer'a göre daha yavaştır, ancak daha iyi tahmin yeteneklerine sahiptir.

Enerji tüketim verilerinde, Encoder-Decoder Transformer, enerji kayıplarının yeniden yapılandırma ve tahmini için etkili bir araçtır. Bu, enerji tüketim seviyelerindeki anomalilerin belirlenmesinde ve enerji verimliliğinin artırılmasında önemli bir rol oynayabilir.

13) **Temporal Fusion Transformers (TFT):** Temporal Fusion Transformers (TFT), özellikle zaman serisi tahmini problemleri için tasarlanmış, dikkat mekanizmasını (attention mechanism) kullanan bir derin öğrenme modelidir. TFT, zaman serisi verilerindeki uzun vadeli bağımlılıkları ve kısa vadeli dinamikleri aynı anda öğrenerek güçlü tahmin performansı sunar.

TFT, zaman serisi tahmini problemlerine özel olarak aşağıdaki özelliklere sahiptir:

- **Çoklu Veriye Uyum:** Hem statik (zamandan bağımsız) hem de dinamik (zamana bağlı) özellikleri işler.
- **Geriye ve İleriye Dönük Dikkat Mekanizması:** Geçmiş ve gelecekteki bilgilere aynı anda odaklanabilir.
- **Değişken Seçimi:** Model, zaman serisi verilerinde hangi değişkenlerin tahmin için daha önemli olduğunu öğrenir.
- **Dikkat Mekanizması:** Hangi zaman adımlarının model için daha kritik olduğunu öğrenerek tahminleri güçlendirir.

TFT modeli, şu ana bileşenlerden oluşur:

- 1) **Değişken Seçimi Katmanları:** Statik ve zamana bağlı değişkenlerden tahmin için en önemli olanları seçer.
- 2) **LSTM Katmanları:** Zaman serisi verilerindeki ardışık bağımlılıkları öğrenir.
- 3) **Dikkat Mekanizması:** Hangi zaman adımlarının tahmin için önemli olduğunu belirler. Çoklu-başlı dikkat mekanizması şu şekilde tanımlanır:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^\top}{\sqrt{d_k}} \right) V \quad (22)$$

- 4) ****Giriş ve Çıkış Katmanları:**** Modelin girdiği veriyi işler ve çıktı olarak tahminleri üretir.

TFT'nin dikkat mekanizması, hangi zaman adımlarının tahmin için önemli olduğunu öğrenmek amacıyla kullanılır. Bu mekanizma, zaman serisi verilerindeki uzun vadeli bağımlılıkları öğrenmek için özellikle etkilidir. Dikkat mekanizması yukarıdaki denklemle ifade edilmiştir.

Temporal Fusion Transformers, zaman serisi verilerinde aşağıdaki avantajları sunar:

- **Uzun ve Kısa Vadeli Dinamikleri Öğrenme:** Hem uzun vadeli trendleri hem de kısa vadeli değişimleri aynı anda modelleyebilir.
- **Değişken Seçimi:** Yüksek boyutlu veri setlerinde önemli değişkenleri otomatik olarak seçer.
- **İnterpretasyon:** Hangi değişkenlerin ve zaman adımlarının tahmin üzerinde daha fazla etkili olduğunu açıklayabilir.
- **Esneklik:** Çoklu veri tipleri (statik ve dinamik) ile çalışabilir.

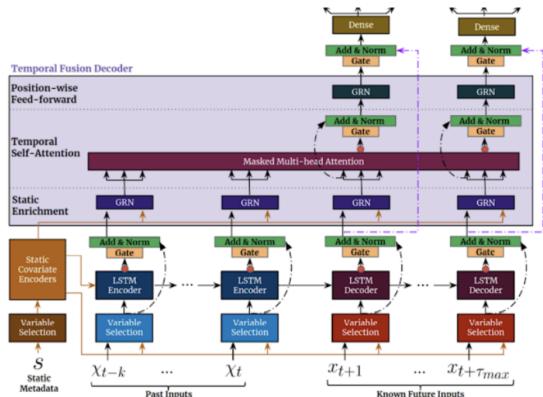


Fig. 29: Temporal Fusion Transformers (TFT) model yapısının genel bir şeması.

Temporal Fusion Transformers (TFT), zaman serisi tahmini için modern bir derin öğrenme modelidir. Dikkat mekanizması, değişken seçimi ve LSTM katmanlarının birleşimi sayesinde güçlü ve esnek tahmin performansı sunmaktadır. Ayrıca, değişkenlerin ve zaman adımlarının önemini açıklayabilmesiyle, model şeffaflık ve yorumlanabilirlik avantajı sağlamaktadır.

Özetle TFT, özellikle zaman serisi tahmini ve açıklanabilirlik için tasarlanmıştır. Geçmiş, gelecekteki

değişkenler ve statik bilgilerle çalışarak daha derin analiz imkanı sağlar. Model, LSTM, attention mekanizmaları ve ek bileşenlerden oluşur. Zaman serisi verilerinin doğasını en iyi şekilde anlamak için karmaşık bir yapı sunar.

Zaman serileri için çok iyi bir uygunluk sunar ve özellikle bu alanda performansı yüksektir. Hem statik hem de dinamik değişkenleri işleyebilir. Çoklu giriş değişkenleri ile çalışabilir. Açıklanabilirlik seviyesi yüksektir ve hangi değişkenlerin sonuçları nasıl etkilediğini anlamak için güçlü bir altyapı sunar.

Performans açısından diğer modellere göre daha yavaştır, ancak güçlü ve detaylı tahmin yeteneklerine sahiptir. TFT, zaman serisi tahmini için özel olarak geliştirilmiş ve açıklanabilirliği yüksek bir modeldir. Özellikle enerji yönetimi gibi zaman serisine dayalı problemlerde büyük avantaj sağlar.

Enerji tüketim verilerinde, TFT, hem statik hem de dinamik değişkenleri analiz ederek enerji tüketim trendlerini anlamada kullanılabılır. Ayrıca, enerji yönetimi süreçlerinde hangi değişkenlerin en önemli olduğunu belirleyerek operasyonel verimliliği artırabilir.

B. Model Değerlendirmeleri Değerlendirme

Bu bölümde, kullanılan modellerin enerji tüketim verilerindeki performanslarının karşılaştırılmalı bir değerlendirmesi yapılacaktır. Çalışma kapsamında kullanılan veri setleri üzerinde makine öğrenmesi, derin öğrenme ve transformatör algoritmalarının sınanması sürecinde doğruluk, açıklanabilirlik, işlem süresi ve kaynak gereksinimleri gibi ölçüm birimleri esas alınmıştır. Bu ölçümlerden elde edilen sonuçlar doğrultusunda, veri setleri üzerinde en uygun değerlendirmeyi yapan öğrenme algoritmalarına karar verilecektir. Her bir modelin avantajları, sınırlılıkları ve uygulama alanındaki etkileri ele alınarak enerji yönetimi süreçlerine olan katkıları analiz edilecektir.

Model değerlendirmesi sürecinde, modelin performansını adil ve doğru bir şekilde ölçebilmek için **veri seti** genellikle eğitim ve test olmak üzere ikiye ayrılır. Bu çalışmada, kullanılan veri setinin %80'i modelin eğitimi için kullanılmış, kalan **%20'lik kısmı ise test verisi** olarak ayrılmıştır. Eğitim verisi, modelin öğrenme sürecinde parametrelerini optimize etmesi ve veriye uyum sağlaması amacıyla kullanılırken, test verisi modelin **görmemişti yeni veriler** üzerindeki performansını değerlendirmek için kullanılmıştır. Bu yöntem sayesinde, modelin aşırı öğrenme (**overfitting**) riskini minimize etmek ve gerçek dünya senaryolarına ne derece uyum sağladığını gözlelemek amaçlanmıştır. Test verisi üzerindeki doğruluk, kesinlik ve duyarlılık gibi performans metrikleri kullanılarak modelin sınıflandırma kabiliyeti detaylı bir şekilde analiz edilmiştir.

1) **Karmaşıklık Matrisi:** Karmaşıklık matrisi, öğrenme modellerinin performanslarının görselleştirildiği bir düzen olarak kullanılmaktadır. Herhangi bir algoritma ile sınıflandırma işlemi yapabilmek için oluşturulmuş olan öğrenme modellerinin, gerçek sınıf değerleri bilinen test verileri üzerinde sınanmaları sonucunda, modelin başarısını gösterir.

Karmaşıklık matrisi tarafından model performansının belirlenmesi amacıyla yapılan tanımlamalar şu şekildedir:

- True Positive (TP):** Gerçek Pozitif - Gerçekte 1 olan sınıf etiketlerinin, 1 olarak tahmin edilme sayısı.
- True Negative (TN):** Gerçek Negatif - Gerçekte 1 olmayan sınıf etiketlerinin, 1 olarak tahmin edilmemesi.
- False Positive (FP):** Yanlış Pozitif - Gerçekte 1 olmayan sınıf etiketlerinin, 1 olarak tahmin edilmesi.
- False Negative (FN):** Yanlış Negatif - Gerçekte 1 olan sınıf etiketlerinin, 1 olarak tahmin edilmemesi.

Karmaşıklık matrisi kullanılarak modellerin doğruluk ve yanlışlık oranları analiz edilir. Bu sayede modelin hata yaptığı durumlar, başarılı tahmin ettiği sınıflar ve performans iyileştirme stratejileri belirlenerek güçlü ve zayıf yönleri ortaya konur. **Derin öğrenme ve transformatör** tabanlı modellerin performansını değerlendirmek için eşik (**threshold**) değerleri belirlenmiştir. Threshold seçiminde öncelikli kriter olarak **F1** skoru kullanılmış, F1 skorlarının eşit olduğu durumlarda ise **precision** (kesinlik) değeri dikkate alınmıştır. Bu doğrultuda, her model için en iyi threshold değeri tespit edilmiş ve parantez içinde ilgili değer belirtilmiştir. Bu yaklaşım, modellerin sınıflandırma performansını optimize etmeyi ve en yüksek başarı oranını elde etmeyi amaçlamaktadır.

TABLE IV: Anomali Optimize Veri Seti (TP, TN, FP, FN)

Model(Varsa Threshold)	TP	TN	FP	FN
NaiveBayes	444	1623	1416	465
LogisticRegression	33	2993	18	904
DecisionTree	172	2938	73	765
RandomForest	91	3001	10	846
KNN	478	2723	316	431
SWM	0	3011	0	937
Autoencoder(0.0)	484	2063	956	424
CNN(0.3)	680	2926	93	228
RNN(0.4)	678	2961	58	230
LSTM(0.3)	704	2950	69	204
GRU(0.4)	835	2959	60	73
Vanilla Transformer(0.3)	848	268	620	67
Encoder-Decoder Transformer(0.1)	903	0	3024	0
Temporal Fusion Transformers (TFT) (0.1)	449	1464	1560	454

TABLE V: Anomali Standart Veri Seti (TP, TN, FP, FN)

Model(Varsa Threshold)	TP	TN	FP	FN
NaiveBayes	732	1664	1986	396
LogisticRegression	30	3614	5	1129
DecisionTree	45	3610	9	1114
RandomForest	66	3616	3	1093
KNN	620	3279	371	508
SWM	0	3619	0	1159
Autoencoder(0.0)	524	2896	751	558
CNN(0.7)	849	3574	73	233
RNN(0.3)	894	3591	56	188
LSTM(0.3)	853	3574	73	229
GRU(0.5)	961	3610	37	121
Vanilla Transformer(0.3)	524	2446	573	384
Encoder-Decoder Transformer(0.1)	1134	0	3595	0
Temporal Fusion Transformers (TFT) (0.1)	511	1971	1624	623

TABLE VI: UNDERSAMPLE Optimize Veri Seti (TP, TN, FP, FN)

Model(Varsa Threshold)	TP	TN	FP	FN
NaiveBayes	458	518	405	427
LogisticRegression	579	560	302	367
DecisionTree	750	485	377	196
RandomForest	673	578	284	273
KNN	676	604	319	209
SWM	537	564	298	409
Autoencoder(0.0)	845	154	734	70
CNN(0.4)	710	800	88	205
RNN(0.4)	738	816	72	177
LSTM(0.3)	755	673	215	160
GRU(0.5)	819	818	70	96
Vanilla Transformer(0.3)	842	287	601	73
Encoder-Decoder Transformer(0.1)	904	0	899	0
Temporal Fusion Transformers (TFT)(0.1)	868	36	863	36

TABLE VII: UNDERSAMPLE Standart Veri Seti (TP, TN, FP, FN)

Model(Varsa Threshold)	TP	TN	FP	FN
NaiveBayes	801	458	683	325
LogisticRegression	754	624	471	418
DecisionTree	936	526	569	236
RandomForest	934	578	517	238
KNN	896	761	380	230
SWM	767	591	504	405
Autoencoder(0.0)	1033	313	810	810
CNN((0.50))	943	1038	85	210
RNN (0.50)	923	1077	46	230
LSTM(0.40)	875	1012	111	278
GRU(0.40)	1056	1069	54	97
Vanilla Transformer(0.30)	1069	356	767	57
Encoder-Decoder Transformer(0.10)	1134	0	1142	0
Temporal Fusion Transformers (TFT)(0.10)	1129	11	1131	5

TABLE VIII: SMOTE Optimize Veri Seti (TP, TN, FP, FN)

Model(Varsa Threshold)	TP	TN	FP	FN
NaiveBayes	1536	1637	1383	1580
LogisticRegression	1904	1938	1158	1136
DecisionTree	1978	2276	820	1062
RandomForest	2304	2173	923	736
KNN	2920	2530	490	196
SWM	1803	2015	1081	1237
Autoencoder(0.0)	2988	0	3059	0
CNN(0.30)	2635	2868	191	353
RNN(0.70)	2417	2906	153	571
LSTM(0.50)	2660	2935	124	328
GRU(0.60)	2859	3009	50	129
Vanilla Transformer(0.4)	2695	1859	1200	293
Encoder-Decoder Transformer(0.1)	3023	0	3024	0
Temporal Fusion Transformers (TFT)(0.1)	2268	799	2225	755

TABLE IX: SMOTE Standart Veri Seti (TP, TN, FP, FN)

Model(Varsa Threshold)	TP	TN	FP	FN
NaiveBayes	2325	1724	1885	1347
LogisticRegression	2468	2089	1567	1157
DecisionTree	2909	2043	1613	716
RandomForest	2969	2209	1447	656
KNN	3399	3095	514	273
SWM	2721	1877	1779	904
Autoencoder(0.0)	3559	0	3631	0
CNN(0.5)	3227	3409	222	332
RNN(0.4)	3119	3389	242	440
LSTM(0.5)	3121	3460	171	438
GRU(0.6)	3305	3542	89	254
Vanilla Transformer(0.4)	3258	1646	1985	301
Encoder-Decoder Transformer(0.1)	3595	0	3595	0
Temporal Fusion Transformers (TFT)(0.1)	2552	1036	2559	1043

C. Ölçümlerin Analizi

Bu bölümde, makine öğrenimi modellerinin performanslarını değerlendirmek için kullanılan ölçütler tanımlanmıştır. Bu ölçütler doğruluk, kesinlik, duyarlılık, F-ölçütü ve hata oranı gibi metrikleri kapsamaktadır. Modellerin HRSS veri setindeki **anomali ölçüm başarıları**, bu ölçütler doğrultusunda detaylı bir şekilde analiz edilmiştir. Yapılan analizler sonucunda, modellerin güçlü ve zayıf yönleri ortaya konmuş, farklı algoritmaların anomali tespitindeki performansları karşılaştırmalı olarak incelenmiştir. Bu değerlendirmeler, hangi modelin belirli senaryolarda daha başarılı olduğunu belirlemesi ve performans iyileştirme stratejilerinin geliştirilmesi açısından önem taşımaktadır.

- **Model Doğruluğu:** Öğrenme modelinin doğruluk derecesini belirler.

$$\text{Doğruluk} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

- **Model Kesinliği:** Öğrenme modelinin duyarlılık derecesini ölçer.

$$\text{Kesinlik} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (9)$$

- **Modelin Duyarlılığı:** Test sonucunda gerçek pozitif değerlerin oranı olarak bilinmektedir.

$$\text{Duyarlılık} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **F Ölçütü:** Duyarlılık ve Kesinlik değerlerinin harmonik ortalamasıdır.

$$F \text{ Ölçütü} = \frac{2 \times \text{Kesinlik} \times \text{Duyarlılık}}{\text{Kesinlik} + \text{Duyarlılık}}$$

- **Hata Oranı:** Hatalı ölçümlerin, toplam değer sayısına olan oranının ölçümüdür.

$$\text{Hata Oranı} = \frac{\text{FP} + \text{FN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

TABLE X: Anomaly Optimize Veri Seti (Accuracy, Precision, Recall, F1 Score, Error Rate)

Model(Varsa Threshold)	Accuracy	Precision	Recall	F1 Score	Error Rate
NaiveBayes	0.5235	0.2387	0.4884	0.3206	0.4764
LogisticRegression	0.7664	0.6470	0.0352	0.0668	0.2335
DecisionTree	0.7877	0.7020	0.1835	0.2910	0.2122
RandomForest	0.7831	0.9009	0.0971	0.1753	0.2168
KNN	0.8107	0.6020	0.5258	0.5613	0.1892
SWM	0.7626	-	0.0	-	0.2373
Autoencoder(0.0)	0.6486	0.3361	0.5330	0.4123	0.3514
CNN(0.3)	0.9183	0.8797	0.7489	0.8090	0.0817
RNN(0.4)	0.9267	0.9212	0.7467	0.8248	0.0733
LSTM(0.3)	0.9305	0.9107	0.7753	0.8376	0.0695
GRU(0.4)	0.9661	0.9330	0.9196	0.9262	0.0339
TF(0.3)	0.6190	0.5777	0.9268	0.7117	0.3810
ED-TF(0.1)	0.2299	0.2299	1	0.3739	0.7701
TFT(0.1)	0.4871	0.2235	0.4972	0.3084	0.5129

TABLE XI: Anomaly Standard Veri Seti (Accuracy, Precision, Recall, F1 Score, Error Rate)

Model(Varsa Threshold)	Accuracy	Precision	Recall	F1 Score	Error Rate
NaiveBayes	0.5014	0.2693	0.6489	0.3806	0.4985
LogisticRegression	0.7626	0.8571	0.0258	0.0502	0.2373
DecisionTree	0.7649	0.8333	0.0388	0.0741	0.2350
RandomForest	0.7706	0.9565	0.0569	0.1074	0.2293
KNN	0.8160	0.6256	0.5496	0.5851	0.1839
SWM	0.7574	-	0.0000	-	0.2425
Autoencoder(0.0)	0.7232	0.4110	0.4842	0.4446	0.2768
CNN(0.7)	0.9353	0.9208	0.7847	0.8473	0.0647
RNN(0.3)	0.9484	0.9411	0.8262	0.8799	0.0516
LSTM(0.3)	0.9361	0.9212	0.7884	0.8496	0.0639
GRU(0.5)	0.9666	0.9629	0.8882	0.9240	0.0334
TF(0.3)	0.7563	0.4777	0.5771	0.5227	0.2437
ED-TF(0.1)	0.2398	0.2398	1	0.3868	0.7602
TFT(0.1)	0.5248	0.2393	0.4506	0.3126	0.4752

TABLE XII: UNDERSAMPLE Optimize Veri Seti (Accuracy, Precision, Recall, F1 Score, Error Rate)

Model(Varsa Threshold)	Accuracy	Precision	Recall	F1 Score	Error Rate
NaiveBayes	0.5398	0.5307	0.5175	0.5240	0.4601
LogisticRegression	0.6299	0.6572	0.6120	0.6338	0.3700
DecisionTree	0.6830	0.6654	0.7928	0.7235	0.3169
RandomForest	0.6919	0.7032	0.7114	0.7073	0.3080
KNN	0.7079	0.6793	0.7638	0.7191	0.2920
SWM	0.6089	0.6431	0.5676	0.6030	0.3910
Autoencoder(0.0)	0.5541	0.5351	0.9235	0.6776	0.4459
CNN(0.4)	0.8375	0.8897	0.7760	0.8290	0.1625
RNN(0.4)	0.8619	0.9111	0.8066	0.8557	0.1381
LSTM(0.3)	0.7920	0.7784	0.8251	0.8011	0.2080
GRU(0.5)	0.9079	0.9213	0.8951	0.9080	0.0921
TF(0.3)	0.6262	0.5835	0.9202	0.7142	0.3738
ED-TF(0.1)	0.5014	0.5014	1	0.6679	0.4986
TFT(0.1)	0.5014	0.5014	0.9602	0.6588	0.4986

TABLE XIII: UNDERSAMPLE Standart Veri Seti (Accuracy, Precision, Recall, F1 Score, Error Rate)

Model(Varsa Threshold)	Accuracy	Precision	Recall	F1 Score	Error Rate
NaiveBayes	0.5553	0.5397	0.7113	0.6137	0.4446
LogisticRegression	0.6078	0.6155	0.6433	0.6291	0.3921
DecisionTree	0.6449	0.6219	0.7986	0.6992	0.3550
RandomForest	0.6669	0.6436	0.7969	0.7121	0.3330
KNN	0.7309	0.7021	0.7957	0.7460	0.2690
SWM	0.5990	0.6034	0.6544	0.6279	0.4009
Autoencoder(0.0)	0.5914	0.5605	0.8959	0.6896	0.4086
CNN(0.5)	0.8704	0.9173	0.8179	0.8647	0.1296
RNN(0.5)	0.8787	0.9525	0.8005	0.8699	0.1213
LSTM(0.4)	0.8291	0.8874	0.7589	0.8181	0.1709
GRU(0.4)	0.9337	0.9514	0.9159	0.9333	0.0663
TF(0.3)	0.6380	0.5883	0.9506	0.7268	0.3620
ED-TF(0.1)	0.4982	0.4982	1	0.6651	0.5018
TFT(0.1)	0.5009	0.4996	0.9956	0.6653	0.4991

TABLE XIV: SMOTE Optimize Veri Seti (Accuracy, Precision, Recall, F1 Score, Error Rate)

Model(Varsa Threshold)	Accuracy	Precision	Recall	F1 Score	Error Rate
NaiveBayes	0.5171	0.5262	0.4929	0.5090	0.4828
LogisticRegression	0.6261	0.6261	0.6263	0.6240	0.3738
DecisionTree	0.6932	0.7069	0.6506	0.6776	0.3067
RandomForest	0.7296	0.7139	0.7578	0.7352	0.2703
KNN	0.8882	0.8563	0.9370	0.8948	0.1117
SWM	0.6222	0.6251	0.5930	0.6087	0.3777
Autoencoder(0.0)	0.4941	0.4941	1	0.6614	0.5059
CNN(0.3)	0.9100	0.9324	0.8819	0.9064	0.0900
RNN(0.7)	0.8803	0.9405	0.8089	0.8697	0.1197
LSTM(0.5)	0.9253	0.5595	0.8902	0.9217	0.0747
GRU(0.6)	0.9704	0.9828	0.9568	0.9696	0.0296
TF(0.4)	0.7531	0.6919	0.9019	0.7831	0.2469
ED-TF(0.1)	0.4999	0.4999	1	0.6666	0.5001
TFT(0.1)	0.5072	0.5048	0.7502	0.6035	0.4928

TABLE XV: SMOTE Standart Veri Seti (Accuracy, Precision, Recall, F1 Score, Error Rate)

Model(Varsa Threshold)	Accuracy	Precision	Recall	F1 Score	Error Rate
NaiveBayes	0.5561	0.5522	0.6331	0.5899	0.4438
LogisticRegression	0.6258	0.6116	0.6808	0.6443	0.3741
DecisionTree	0.6801	0.6432	0.8024	0.7141	0.3198
RandomForest	0.7111	0.6723	0.8190	0.7384	0.2888
KNN	0.8919	0.8686	0.9256	0.8962	0.1080
SWM	0.6315	0.6046	0.7506	0.6697	0.3684
Autoencoder(0.0)	0.4950	0.4950	1	0.6614	0.5059
CNN(0.5)	0.9229	0.9356	0.9067	0.9209	0.0771
RNN(0.4)	0.9051	0.9280	0.8764	0.9014	0.0949
LSTM(0.5)	0.9153	0.9481	0.8769	0.9111	0.0847
GRU(0.6)	0.9523	0.9738	0.9286	0.9507	0.0477
TF(0.4)	0.6821	0.6214	0.9154	0.7403	0.3179
ED-TF(0.1)	0.5000	0.5000	1	0.6667	0.5000
TFT(0.1)	0.4990	0.4993	0.7099	0.5863	0.5010

Model Performans Değerlendirmesi

Bu çalışmada, çeşitli makine öğrenimi ve derin öğrenme modellerinin üç farklı veri seti (Anomali, Undersample ve SMOTE) üzerindeki doğruluk ve F1 Score performansları analiz edilmiştir. Amaç, bu modeller arasında en iyi performansı gösteren modeli seçmek ve bu seçimin nedenlerini açık bir şekilde açıklamaktır.

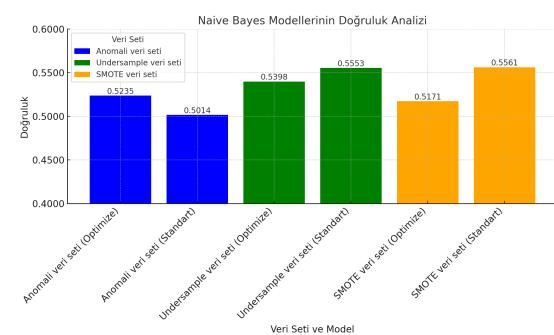


Fig. 30: Naive Bayes Modellerinin Doğruluk Karşılaştırması

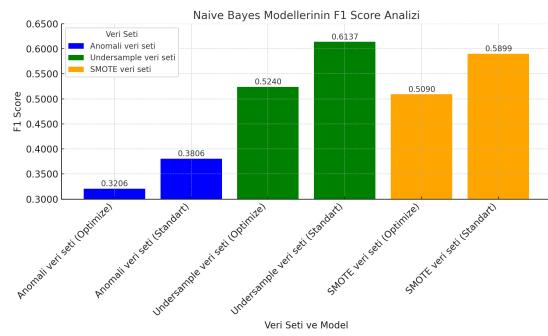


Fig. 31: Naive Bayes Modellerinin F1 Skor Karşılaştırması

1) *Naive Bayes*: Naive Bayes modellerinin F1 skor ve doğruluk analizlerine baktığımızda, farklı veri setlerinin modeli üzerindeki etkileri açıkça görülmektedir. SMOTE ile dengelenmiş veri seti, optimize edilmiş ve standart versiyonlarıyla birlikte, F1 skorunda ve doğruluk oranında en yüksek performansı sergilemiştir. SMOTE veri setiyle elde edilen F1 skoru 0.5899'a ulaşırken doğruluk oranı 0.5561 ile diğer yöntemleri geride bırakmıştır. Öte yandan, anomalili veri seti hem optimize edilmiş hem de standart versiyonunda en düşük performansı göstermiştir. Bu sonuçlar, veri dengesizliği problemini çözmek için SMOTE yönteminin etkili bir yaklaşım olduğunu ve model başarısını artırmada kritik bir rol oynadığını ortaya koymaktadır.

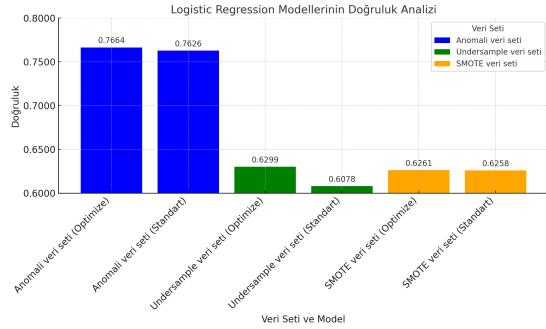


Fig. 32: Logistic Regression Modellerinin Doğruluk Karşılaştırması

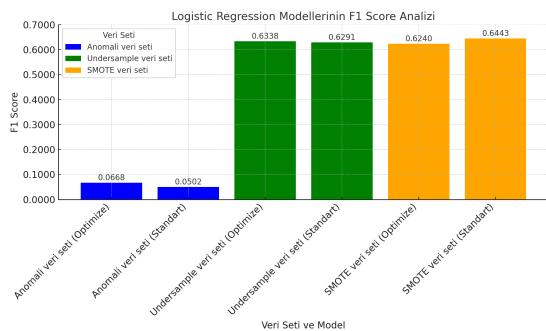


Fig. 33: Logistic Regression Modellerinin F1 Skor Karşılaştırması

2) *Logistic Regression*: Logistic Regression modellerinin F1 skor ve doğruluk analizlerine baktığımızda, anomalili veri seti optimize edilmiş ve standart versiyonlarında en yüksek doğruluk oranlarını (sırasıyla 0.7664 ve 0.7626) sağlamıştır. Ancak F1 skoru açısından anomalili veri seti en düşük değerleri (0.0668 ve 0.0502) göstermiştir. SMOTE ve Undersample yöntemleri, F1 skor performansında dengeli veri seti oluşturanın önemini vurgulamaktadır. SMOTE yöntemi, standart versiyonunda 0.6443 ile en iyi F1 skoruna ulaşırken doğruluk oranı 0.6258'dir. Bu sonuçlar, model performansını artırmada veri dengesizliği ile başa çıkmının önemli olduğunu göstermektedir.

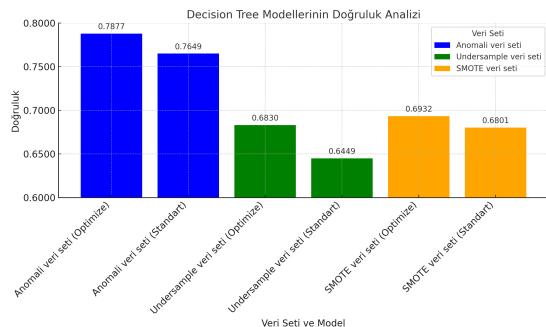


Fig. 34: Decision Tree Modellerinin Doğruluk Karşılaştırması

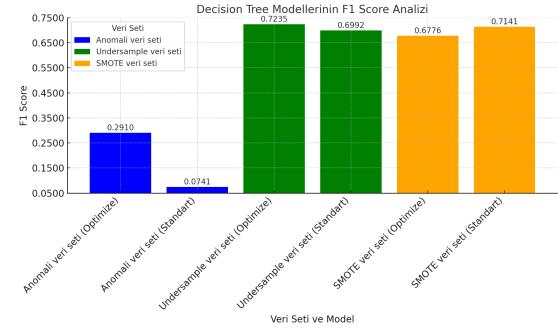


Fig. 35: Decision Tree Modellerinin F1 Skor Karşılaştırması

3) *Decision Tree*: Decision Tree modellerinin F1 skor ve doğruluk analizlerine baktığımızda, anomalili veri seti optimize edilmiş ve standart versiyonlarında en yüksek doğruluk oranlarını (sırasıyla 0.7877 ve 0.7649) sağlamıştır. Ancak, F1 skoru açısından anomalili veri seti en düşük değerleri (0.2910 ve 0.0741) göstermiştir. Undersample yöntemi optimize edilmiş versiyonunda en yüksek F1 skorunu (0.7235) sağlamış ve SMOTE yöntemi ise standart versiyonunda 0.7141 ile benzer bir başarı göstermiştir. Bu sonuçlar, dengesiz veri setlerinin model performansı üzerindeki etkisini ve farklı dengeleme yöntemlerinin etkili kullanımını vurgulamaktadır.

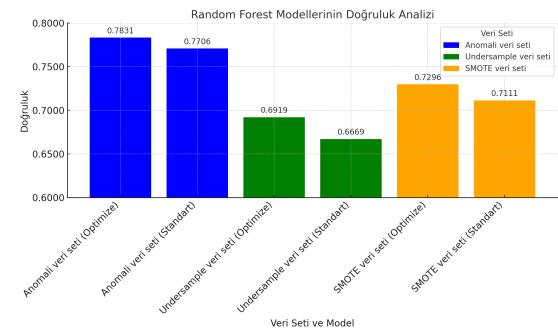


Fig. 36: Random Forest Modellerinin Doğruluk Karşılaştırması

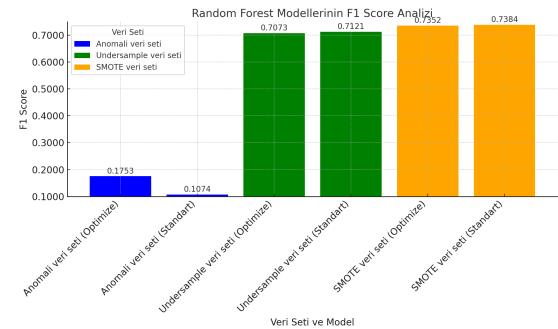


Fig. 37: Random Forest Modellerinin F1 Skor Karşılaştırması

4) *Random Forest*: Random Forest modellerinin F1 skor ve doğruluk analizlerine baktığımızda, anomalili veri seti optimize edilmiş ve standart versiyonlarında en yüksek doğruluk

oranlarını (sırasıyla 0.7831 ve 0.7706) sağlamıştır. Ancak F1 skoru açısından anomalili veri seti en düşük değerleri (0.1753 ve 0.1074) göstermiştir. SMOTE yöntemi standart versiyonunda 0.7384 F1 skoru ile en yüksek başarıyı elde etmiş ve doğruluk oranında da 0.7111 seviyesine ulaşmıştır. Undersample yöntemi ise optimize edilmiş versiyonunda 0.7121 F1 skoru ile etkili bir performans göstermiştir. Bu sonuçlar, dengesiz veri setlerinin etkisinin Random Forest model performansında açıkça görüldüğünü ve SMOTE ile dengeli veri setlerinin model başarısını artırmada önemli bir araç olduğunu göstermektedir.

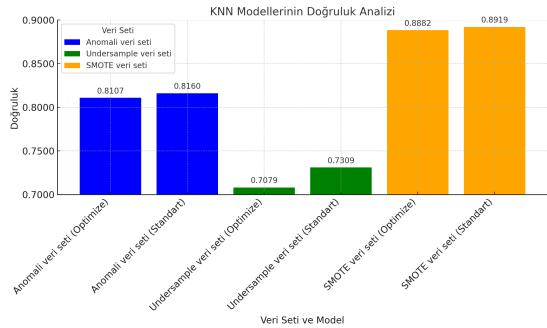


Fig. 38: KNN Modellerinin Doğruluk Karşılaştırması

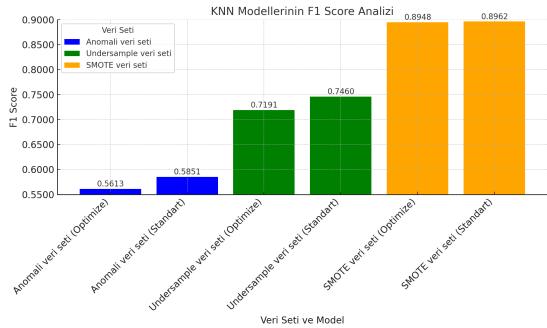


Fig. 39: KNN Modellerinin F1 Skor Karşılaştırması

5) *K-Nearest Neighbors (KNN)*: KNN modellerinin F1 skor ve doğruluk analizlerine baktığımızda, SMOTE yöntemiyle dengelenmiş veri seti, optimize edilmiş ve standart versiyonlarında hem F1 skoru hem de doğruluk oranında en yüksek performansı göstermiştir. SMOTE yönteminde F1 skoru 0.8962'ye, doğruluk oranı ise 0.8919'a ulaşmıştır. Anomalili veri seti, optimize edilmiş ve standart versiyonlarında doğruluk oranında yüksek değerler (0.8107 ve 0.8160) elde etse de, F1 skoru açısından daha düşük değerler (0.5613 ve 0.5851) göstermiştir. Undersample yöntemi ise optimize edilmiş versiyonunda 0.7191 F1 skoru ve 0.7079 doğruluk oranı ile dengeli bir performans sağlamıştır. Bu sonuçlar, veri dengesizliğini ele almanın KNN model performansı üzerindeki etkisini ve SMOTE yönteminin etkili bir yaklaşım olduğunu göstermektedir.

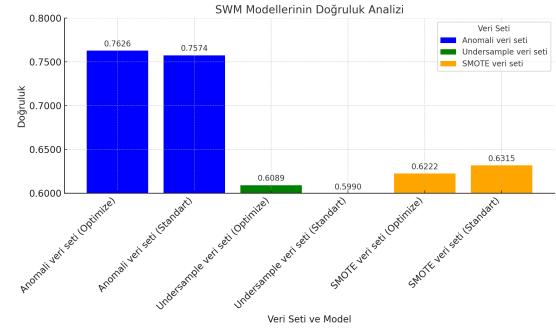


Fig. 40: SVM Modellerinin Doğruluk Karşılaştırması

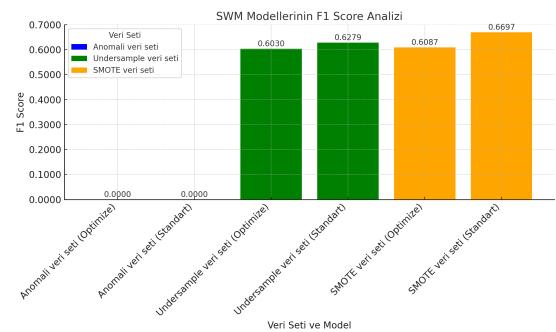


Fig. 41: SVM Modellerinin F1 Skor Karşılaştırması

6) *Support Vector Machine (SVM)*: Support Vector Machine (SVM) modellerinin F1 skor ve doğruluk analizlerinde, anomalili veri seti optimize edilmiş ve standart versiyonlarında doğruluk oranlarında en yüksek değerler (sırasıyla 0.7626 ve 0.7574) elde edilmiştir. Ancak, anomalili veri seti F1 skoru açısından 0.0000 ile performans sergileyememiştir. SMOTE yöntemi, standart versiyonunda 0.6697 F1 skoru ile en yüksek başarıyı göstermiştir. Undersample yöntemi ise optimize edilmiş versiyonunda 0.6279 F1 skoru ile başarılı bir performans sergilemiştir. Bu analizler, SMOTE yönteminin dengeli veri setleri oluşturmada etkili bir yöntem olduğunu ve model başarısını artırmada önemli bir rol oynadığını göstermektedir.

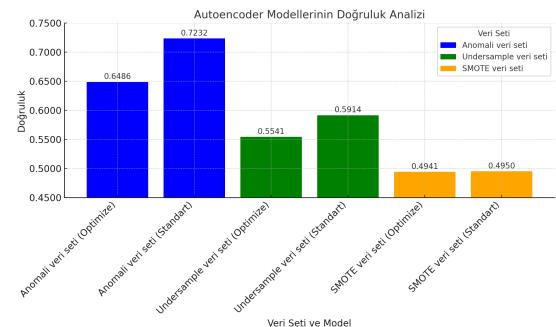


Fig. 42: Autoencoder Modellerinin Doğruluk Karşılaştırması

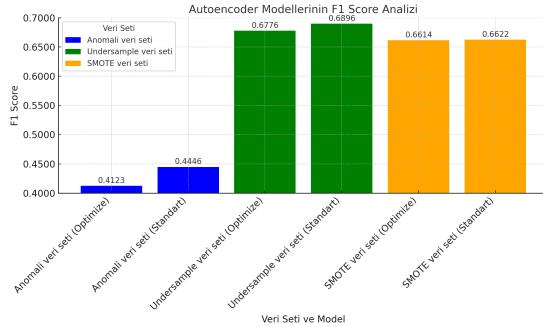


Fig. 43: Autoencoder Modellerinin F1 Skor Karşılaştırması

7) *Autoencoder*: Autoencoder modellerinin F1 skor ve doğruluk analizlerinde, anomalili veri seti standart versiyonunda doğruluk açısından en yüksek değeri (0.7232) sağlamış, ancak F1 skor açısından 0.4446 ile daha düşük bir performans sergilemiştir. Undersample yöntemi, optimize edilmiş versiyonunda 0.6776 F1 skorunu ve standart versiyonunda 0.6896 ile dengeli bir performans göstermiştir. SMOTE yöntemi ise standart versiyonunda 0.6622 F1 skorunu ve 0.4950 doğruluk oranı ile orta seviyede bir başarı sağlamıştır. Bu sonuçlar, veri dengesizliğinin Autoencoder modelleri üzerindeki etkisini ve farklı dengeleme yöntemlerinin önemini göstermektedir.

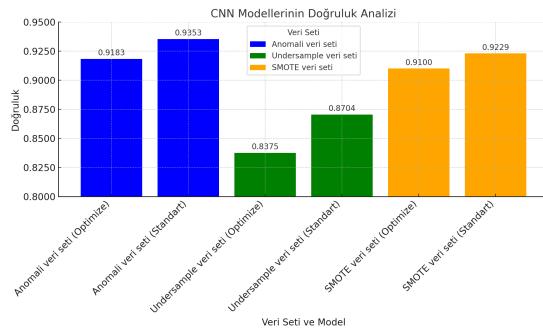


Fig. 44: CNN Modellerinin Doğruluk Karşılaştırması

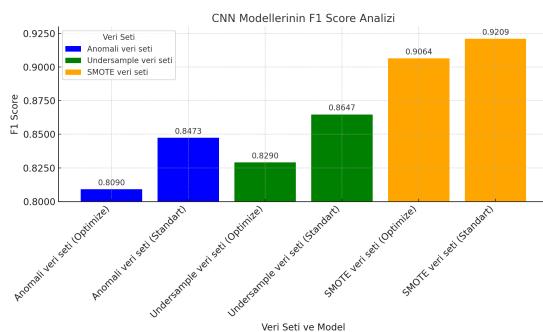


Fig. 45: CNN Modellerinin F1 Skor Karşılaştırması

8) *Convolutional Neural Network (CNN)*: Convolutional Neural Network (CNN) modellerinin doğruluk ve F1 skor analizlerinde, SMOTE yöntemi standart versiyonunda hem

doğruluk (0.9229) hem de F1 skoru (0.9209) açısından en yüksek performansı göstermiştir. Anomalili veri seti standart versiyonunda doğruluk oranında en yüksek değeri (0.9353) sağlamışken, F1 skoru açısından 0.8473 ile nispeten daha düşük bir performans sergilemiştir. Undersample yöntemi ise optimize edilmiş versiyonunda 0.8290 F1 skoru ve 0.8375 doğruluk oranı ile orta düzey bir performans göstermiştir. Bu analizler, SMOTE yönteminin CNN model performansını artırmada etkili bir yöntem olduğunu ve veri dengesizliğiyle başa çıkmada önemli bir araç olduğunu göstermektedir.

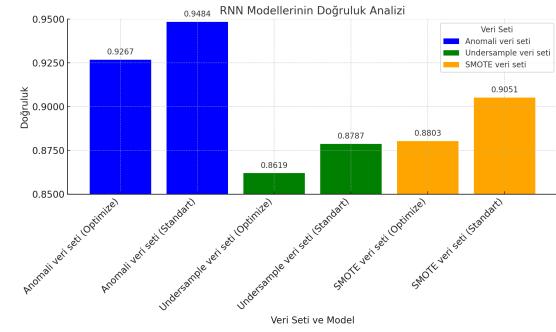


Fig. 46: RNN Modellerinin Doğruluk Karşılaştırması

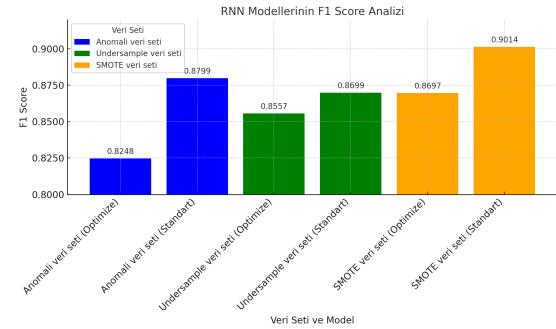


Fig. 47: RNN Modellerinin F1 Skor Karşılaştırması

9) *Recurrent Neural Network (RNN)*: Recurrent Neural Network (RNN) modellerinin doğruluk ve F1 skor analizlerinde, anomalili veri seti standart versiyonunda doğruluk açısından en yüksek değeri (0.9484) sağlamış ve F1 skoru açısından da 0.8799 ile güçlü bir performans göstermiştir. SMOTE yöntemi standart versiyonunda hem doğruluk (0.9051) hem de F1 skoru (0.9014) açısından etkili bir sonuç elde etmiştir. Undersample yöntemi ise optimize edilmiş versiyonunda 0.8619 doğruluk oranı ve 0.8557 F1 skoru ile ortalama bir başarı göstermiştir. Bu analizler, SMOTE yönteminin RNN modellerinde hem doğruluk hem de F1 skor performansını artırmada önemli bir rol oynadığını ve dengeli veri setleri oluşturmanın etkili bir yöntem olduğunu göstermektedir.

10) *Long Short-Term Memory (LSTM)*: Long Short-Term Memory (LSTM) modellerinin doğruluk ve F1 skor analizlerinde, SMOTE yöntemi optimize edilmiş versiyonunda hem

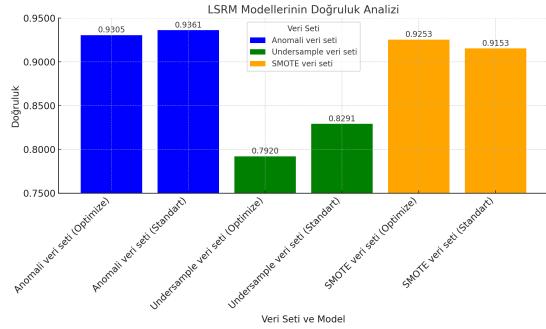


Fig. 48: LSTM Modelleriminin Doğruluk Karşılaştırması

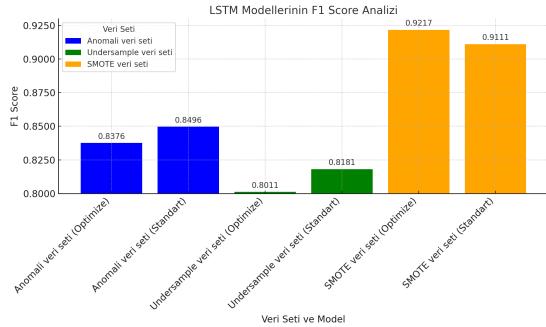


Fig. 49: LSTM Modelleriminin F1 Skor Karşılaştırması

doğruluk (0.9253) hem de F1 skoru (0.9217) açısından en yüksek performansı göstermiştir. Anomalili veri seti standart versiyonunda doğruluk açısından en yüksek değeri (0.9361) sağlamış, F1 skoru açısından ise 0.8496 ile güçlü bir performans sergilemiştir. Undersample yöntemi ise optimize edilmiş versiyonunda 0.8011 F1 skoru ve 0.7920 doğruluk oranı ile nispeten düşük bir başarı göstermiştir. Bu analizler, SMOTE yönteminin LSTM modellerinde dengeli veri setleri oluşturanın doğruluk ve F1 skor performansını artırmada etkili olduğunu ortaya koymaktadır.

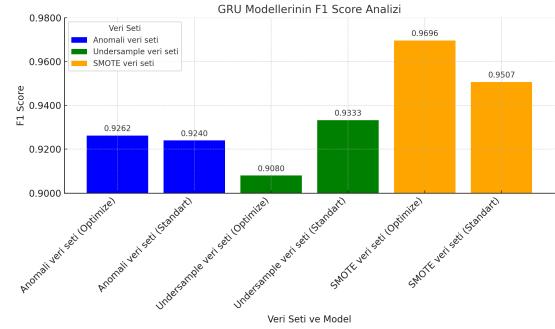


Fig. 51: GRU Modelleriminin F1 Skor Karşılaştırması

11) Gated Recurrent Unit (GRU): Gated Recurrent Unit (GRU) modellerinin doğruluk ve F1 skor analizlerinde, SMOTE yöntemi optimize edilmiş versiyonunda hem doğruluk (0.9704) hem de F1 skoru (0.9696) açısından en yüksek performansı göstermiştir. Anomalili veri seti optimize edilmiş ve standart versiyonlarında sırasıyla 0.9661 ve 0.9666 doğruluk oranı ile güçlü bir performans sergilemiş, F1 skorları ise sırasıyla 0.9262 ve 0.9240 olarak hesaplanmıştır. Undersample yöntemi optimize edilmiş versiyonunda 0.908 doğruluk oranı ve 0.9079 F1 skoru ile nispeten düşük bir başarı sergilemiştir. Bu sonuçlar, SMOTE yönteminin GRU modellerinin performansını artırmada önemli bir araç olduğunu ve veri dengesizliğini başarılı bir şekilde ele aldığı göstermektedir.

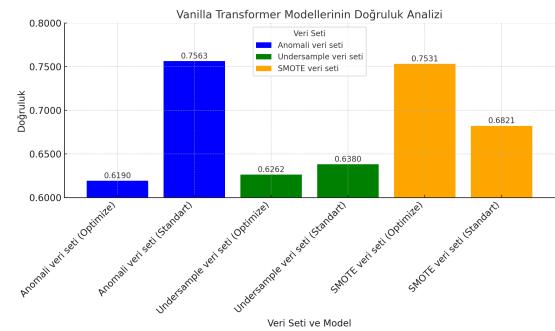


Fig. 52: Vanilla Transformer Modelerinin Doğruluk Karşılaştırması

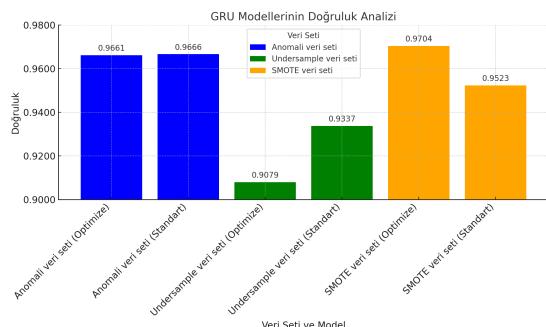


Fig. 50: GRU Modelleriminin Doğruluk Karşılaştırması

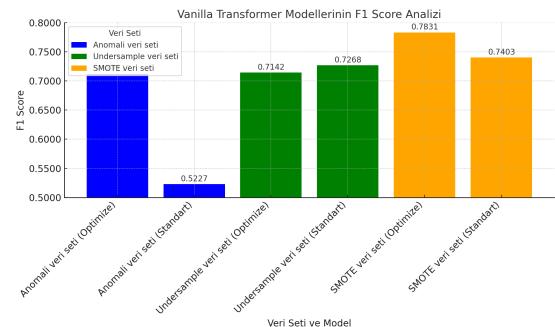


Fig. 53: Vanilla Transformer Modelerinin F1 Skor Karşılaştırması

12) Vanilla Transformer: Vanilla Transformer modellerinin doğruluk ve F1 skor analizlerinde, SMOTE yöntemi optimize edilmiş versiyonunda F1 skor açısından en yüksek değeri (0.7831) sağlamış, doğruluk oranında ise 0.7531 ile güçlü bir performans sergilemiştir. Anomalili veri seti standart versiyonunda doğruluk oranında 0.7563 ile en iyi sonuçları elde etmiş, ancak F1 skor açısından 0.5227 ile nispeten düşük bir performans göstermiştir. Undersample yöntemi standart versiyonunda 0.7268 F1 skoru ile dikkat çekmiş ve optimize edilmiş versiyonunda 0.6380 doğruluk oranı ile orta düzeyde bir başarı sergilemiştir. Bu analizler, SMOTE yönteminin Vanilla Transformer modellerinin başarısını artırmada önemli bir araç olduğunu ortaya koymaktadır.

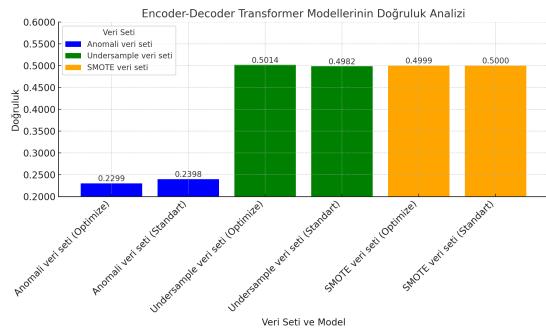


Fig. 54: Encoder-Decoder Transformer Modelerinin Doğruluk Karşılaştırması

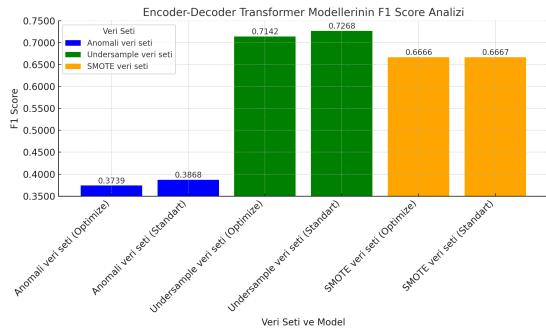


Fig. 55: Encoder-Decoder Transformer Modelerinin F1 Skor Karşılaştırması

13) Encoder-Decoder Transformer: Encoder-Decoder Transformer modellerinin doğruluk ve F1 skor analizlerinde, Undersample yöntemi standart versiyonunda en yüksek F1 skorunu (0.7268) sağlamış ve doğruluk açısından ise 0.5014 ile dikkat çekmiştir. SMOTE yöntemi standart ve optimize edilmiş versiyonlarında benzer doğruluk oranları (0.5000 ve 0.4999) ile orta seviyede bir başarı elde etmiştir. Anomalili veri seti optimize edilmiş ve standart versiyonlarında doğruluk açısından düşük performans göstermiş (0.2299 ve 0.3868), F1 skorunda ise sırasıyla 0.3739 ve 0.3868 ile en düşük sonuçları almıştır. Bu analizler, veri dengesizliğini ele almak için Undersample yönteminin Encoder-Decoder Transformer modellerinde en etkili yöntem olduğunu ortaya koymaktadır.

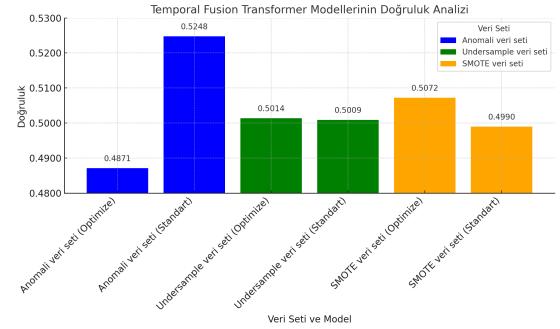


Fig. 56: Temporal Fusion Transformer Modelerinin Doğruluk Karşılaştırması

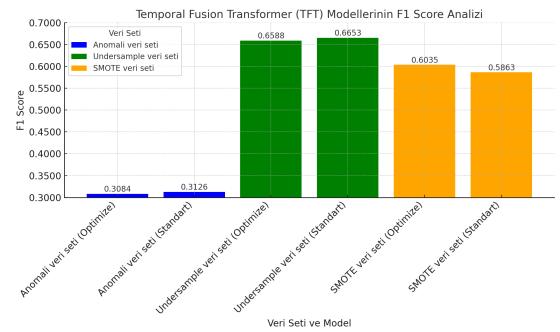


Fig. 57: Temporal Fusion Transformer Modelerinin F1 Skor Karşılaştırması

14) Temporal Fusion Transformer (TFT): Temporal Fusion Transformer (TFT) modellerinin doğruluk ve F1 skor analizlerinde, Undersample yöntemi standart versiyonunda en yüksek F1 skorunu (0.6653) ve doğruluk oranını (0.5014) elde etmiştir. SMOTE yöntemi optimize edilmiş versiyonunda 0.6035 F1 skoru ve 0.5072 doğruluk orANIyla güçlü bir performans sergilemiştir. Anomalili veri seti, optimize edilmiş ve standart versiyonlarında sırasıyla 0.3084 ve 0.3126 F1 skorları ve 0.4871 ile 0.5248 doğruluk oranlarıyla düşük performans göstermiştir. Bu analizler, veri dengeme yöntemlerinin TFT modellerindeki performansı artırmadaki önemini vurgulamaktadır.

Analiz Bulguları

Yapılan analizler sonucunda aşağıdaki gözlemler elde edilmiştir:

- Basit modeller (Naive Bayes, SWM, Decision Tree) özellikle anomalili veri setinde düşük performans sergilemiştir.
- Karmaşık modeller (Random Forest, KNN), dengelenmiş veri setlerinde daha yüksek performans göstermiştir.
- Derin öğrenme modelleri (GRU, LSTM, CNN), özellikle SMOTE ve undersample veri setlerinde en iyi sonuçları vermiştir.
- Transformer tabanlı modeller (Vanilla Transformer, ED-TF, TFT), zamansal karmaşılık içeren veri setleri için

umut vaat etmekle birlikte, diğer derin öğrenme modellerine kıyasla daha düşük performans göstermiştir.

Model Seçimi

Bu çalışmada, çeşitli makine öğrenimi ve derin öğrenme modellerinin üç farklı veri seti (Anomali, Undersample ve SMOTE) üzerindeki doğruluk ve F1 Score performansları analiz edilmiştir. Amaç, bu modeller arasında en iyi performansı gösteren modeli ve veri setini seçmek, bu seçim nedenlerini açık bir şekilde açıklamaktır.

Analiz Bulguları

Yapılan analizler sonucunda aşağıdaki gözlemler elde edilmiştir:

- Basit modeller (Naive Bayes, SWM, Decision Tree) özellikle anomali veri setinde düşük performans sergilemiştir.
- Karmaşık modeller (Random Forest, KNN), dengelenmiş veri setlerinde daha yüksek performans göstermiştir.
- Derin öğrenme modelleri (GRU, LSTM, CNN), özellikle SMOTE ve undersample veri setlerinde en iyi sonuçları vermiştir.
- Transformer tabanlı modeller (Vanilla Transformer, EDTF, TFT), zamansal karmaşıklık içeren veri setleri için umut vaat etmekle birlikte, diğer derin öğrenme modellerine kıyasla daha düşük performans göstermiştir.

Model ve Veri Seti Seçimi

Yapılan analizlerin sonucunda, **GRU (Gated Recurrent Unit)** modeli ve **SMOTE veri seti** en iyi performansı göstermiştir. Bu seçim nedenleri aşağıdaki gibi sıralanabilir:

1. *Genel Performans*: GRU modeli, tüm veri setlerinde tutarlı bir şekilde yüksek doğruluk ve F1 Score değerleri sağlamıştır. Özellikle SMOTE veri setinde elde edilen F1 Score değeri 0.9696 ve doğruluk değeri 0.9704, diğer modellerden üstün olduğunu göstermektedir. SMOTE veri seti, sınıf dengesini sağlamak için kullanılan bir teknik olarak, modelin potansiyelini en üst düzeye çıkarmıştır.

2. *Zamansal Veri İşleme Yeteneği*: GRU, zamansal verilerdeki ilişkileri modellemek için tasarlanmıştır. SMOTE veri seti ile çalışıldığında, veri dengesinin sağlanması GRU'nun kapi mekanizmalarının zaman serisi ilişkilerini daha iyi öğrenmesine olanak tanımıştır. Bu, özellikle SMOTE veri setindeki üstün performansla kanıtlanmaktadır.

3. *Hesaplama Verimliliği*: LSTM modellerine kıyasla GRU, daha az parametreye sahip olması nedeniyle daha düşük hesaplama maliyeti ile çalışmaktadır. Bu durum, büyük ölçüde SMOTE veri setleri üzerinde daha hızlı ve verimli sonuçlar elde edilmesini sağlamıştır.

4. *Veri Dengeleme Teknikleri ile Uyum*: SMOTE veri seti, dengesiz sınıf dağılımlarını gidermiş ve GRU'nun potansiyelini en iyi şekilde ortaya koymasını sağlamıştır. SMOTE ile oluşturulan dengeli veri seti, modelin öğrenme kapasitesini artırarak daha tutarlı ve yüksek performanslı sonuçlar üretmiştir.

Sonuç

Bu çalışmada, GRU modeli ve SMOTE veri seti en iyi genel performansı gösteren kombinasyon olarak seçilmiştir. Modelin zamansal verilerdeki ilişkileri öğrenme kapasitesi, düşük hesaplama maliyeti ve SMOTE veri setinin sınıf dengesini sağlamadaki başarısı, seçimde belirleyici faktörler olmuştur. GRU'nun bu tüstünlükleri, özellikle zaman serisi ve dengesiz veri setleri içeren projelerde SMOTE veri seti ile birlikte kullanılmasını önerir.

Gelecekteki çalışmalarında, GRU'nun hiperparametre optimizasyonu ve daha karmaşık veri kümeleri üzerindeki performansı araştırılabilir. Ayrıca, diğer veri dengeleme tekniklerinin GRU performansına etkisi incelenebilir.

VI. KAFKA VE SPARK ENTEGRASYONU İLE GERÇEK ZAMANLI VERİ İŞLEME VE ANOMALI TESPİTİ

Bu çalışma, HRSS'nin enerji tüketim verilerinin analiz edilmesi, operasyonel süreçlerdeki verimsizliklerin tespit edilmesi ve enerji kayıplarının önlenmesi için yenilikçi bir yaklaşım geliştirmeyi amaçlamaktadır. Veri odaklı yaklaşımlar ve makine öğrenimi teknikleri kullanılarak, enerji yönetimi ve operasyonel süreçlerde karşılaşılan zorluklara çözüm önerileri sunulacaktır. Bununla birlikte, Apache Kafka ve Apache Spark gibi büyük veri teknolojilerinin entegrasyonu ile gerçek zamanlı analiz kabiliyetlerinin artırılması hedeflenmektedir.

A. Apache Kafka

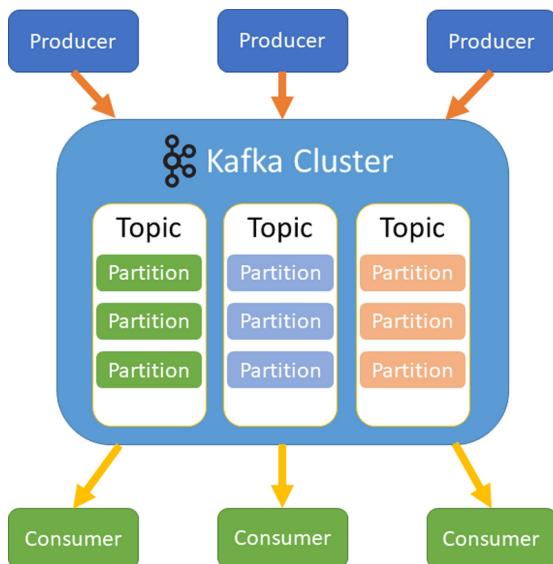


Fig. 58: Kafka Modeli

Apache Kafka, dağıtık bir akış platformudur ve gerçek zamanlı veri akışı (*streaming*) için kullanılan bir mesajlaşma sistemidir. Kafka, veri akışı neredeyse sıfır gecikme ile işleme yeteneği sunarak, hem çeşitli veri kaynaklarından gelen bilgileri toplar hem de bu verileri tüketici uygulamalara iletir. Kafka, yüksek performans, düşük gecikme ve yatayda kolayca skalabilir yapısı sayesinde büyük veri ekosistemlerinde kritik bir rol oynamaktadır.

Kafka'nın temel bileşenleri şunlardır:

- **Producer:** Veriyi Kafka'ya gönderir.
- **Consumer:** Kafka'dan veriyi tüketir.
- **Broker:** Gelen veriyi depolar ve yönetir.
- **Topic:** Verinin kategorilere ayrılması için kullanılır.

Apache Kafka bu projede şu amaçlarla kullanılacaktır:

- Gerçek zamanlı veri akışı yöneterek, enerji tüketim verilerinin anlık olarak izlenmesini sağlamak.
- Operasyonel süreçlerden gelen verileri merkezileştirerek, farklı sistemlerden gelen bilgilerin tutarlığını sağlamak.
- Olası anormalliklerin ve enerji kayıplarının erken tespit edilmesi için bir veri akış hattı oluşturmak.

B. Apache Spark

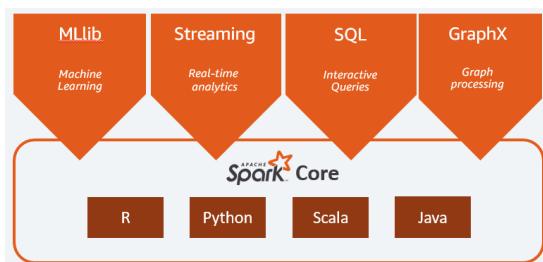


Fig. 59: Apache Temel Bileşenleri

Apache Spark, büyük veri işleme için kullanılan hızlı ve genel bir hesaplama motorudur. Spark, gerçek zamanlı veri akış işleme, toplu işlem (*batch processing*), makine öğrenimi ve grafik analizi gibi çok yönlü işlevler sunar. Hafıza içinde (*in-memory*) hesaplama yapabilme yeteneği sayesinde Spark, veri işleme performansında çarpıcı etkiler yaratmaktadır.

Spark'in temel bileşenleri şunlardır:

- **Spark Core:** Tüm Spark işlevleri için temel altyapı.
- **Spark SQL:** Yapılandırılmış verilerle çalışmak için SQL sorgu yeteneği.
- **Spark Streaming:** Gerçek zamanlı veri akış işleme.
- **MLlib:** Makine öğrenimi kütüphanesi.
- **GraphX:** Grafik ve grafik tabanlı hesaplama.

Apache Spark bu projede şu amaçlarla kullanılacaktır:

- Enerji tüketim verilerinin hızlı bir şekilde analiz edilmesini sağlamak.
- Makine öğrenimi algoritmaları uygulayarak, enerji kayıplarının önlenmesi ve operasyonel iyileştirme için tahminsel modeller oluşturmak.
- Gerçek zamanlı analizlerle anormallik tespitini otomatize etmek ve enerji yönetiminde daha verimli kararlar alınmasını desteklemek.

Apache Kafka ve Spark entegrasyonu, gerçek zamanlı veri akışı çok daha etkin bir şekilde işleme ve analiz etme olanağı sunarak, enerji yönetimi gibi karmaşık sistemlerde verimli çözüm yolları oluşturmayı mümkün kılar.

C. Teknik Mimari ve Adımlar

Bu sistemin teknik mimaris, Kafka ve Spark'in çok yönlü yeteneklerini kullanarak GRU modeline dayalı bir anomali tespiti yapısı sunmaktadır. Aşağıdaki adımlar, sistemin başlıca çalışma prensiplerini detaylandırır:

1) Veri Toplama:

- Enerji tüketim verileri, Kafka Producer bileşenleri aracılığıyla bir Kafka Topic'e gönderilir.
- Veriler JSON formatında gönderilerek, şema bazlı uyumluluk sağlanır.

2) Gerçek Zamanlı Veri İşleme:

- Kafka Consumer, gelen verileri dinler ve Spark Streaming ile entegre edilmiş bir veri hattına iletir.
- Spark Streaming, verileri ön işlemeden geçirir ve GRU modeline uygun bir formata dönüştürür.

3) GRU Modeli ile Anomali Tespiti:

- GRU modeli, Spark MLlib veya TensorFlow ile entegre edilmiş olarak kullanılır.
- Model, gelen verileri analiz eder ve olası anormallikleri tespit eder.
- Anomali skoru, belirlenen bir eşik değerinin üzerindeyse, veri "anormal" olarak işaretlenir.

4) Sonuçların Yayınlanması:

- Tespit edilen anomaliler, Kafka Producer aracılığıyla yeni bir Kafka Topic'e yazılır.
- Alternatif olarak, tespit edilen anomaliler terminale basılır.

5) Sürekliklilik ve Skalabilite:

- Sistem, Kafka'nın yatayda skalabilir yapısı sayesinde artan veri hacimlerine kolayca uyum sağlar.
- Spark Streaming, gerçek zamanlı analizlerde düşük gecikme sağlayarak performansı optimize eder.

Bu mimari, enerji tüketim verilerinin anlık izlenmesini ve olası anormalliklerin erkenden tespit edilmesini sağlayarak enerji yönetiminde etkin bir çözüm sunar.

D. Makine Öğrenimi Tabanlı Anomali Tespiti

Anomali tespit süreci, TensorFlow tabanlı bir model kullanılarak HRSS veri setinin ön işlenmesi ile başlamaktadır. Bu amaçla kullanılan Python betiği şu adımları kapsamaktadır:

- Makine öğrenimi modelinin ve veri setinin yüklenmesi.
- Veri seti özelliklerinin GRU modeli ile uyumlu olacak şekilde yeniden düzenlenmesi.
- Tahmin yapılarak verilerin normal ya da anormal olarak etiketlenmesi.
- Tahmin sonuçlarının bir CSV dosyasına kaydedilmesi.

1) Kod Uygulaması:

Python betiği aşağıdaki komut ile çalıştırılmıştır:

```
python <script_path> <model_path> <data_path>
```

Oluşturulan çıktı CSV dosyası yapısından bir örnek aşağıda verilmiştir:

Ozellik1	...	Etiketler	Tahminler
0.12	...	0	0.85

Bu süreç şekil 60'de gösterilmektedir.

Fig. 60: Anomali tespiti için makine öğrenimi için gönderilen veriler.

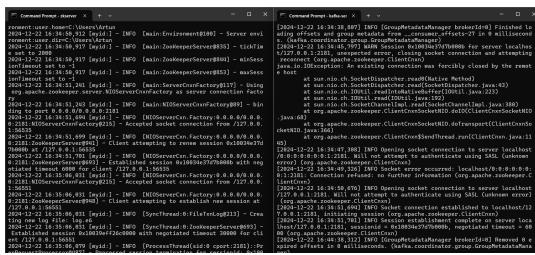


Fig. 61: Topicler için kurlan ortam.

```
[4/17/22 17:06:59 INFO BlockManager] Initiating blockmanager... BlockManager[0]
[4/24/22 17:07:02:09 798447: I tensorflow/core/platform/cpu_feature_guard.cc:194] enable them in other operations, rebuild TensorFlow with the appropriate co
 1/1124 [.....] - ETA: 4:07
29/1124 [.....] - ETA: 2s
57/1124 [>.....] - ETA: 2s
85/1124 [=>.....] - ETA: 2s
116/1124 [==>.....] - ETA: 2s
141/1124 [===>.....] - ETA: 2s
171/1124 [===>.....] - ETA: 1s
204/1124 [===>.....] - ETA: 1s
235/1124 [=====>.....] - ETA: 1s
263/1124 [======>.....] - ETA: 1s
290/1124 [======>.....] - ETA: 1s
320/1124 [======>.....] - ETA: 1s
347/1124 [======>.....] - ETA: 1s
381/1124 [======>.....] - ETA: 1s
411/1124 [======>.....] - ETA: 1s
436/1124 [======>.....] - ETA: 1s
```

Fig. 62: Modelin verileri gönderilme aşaması

E. Apache Kafka ve Spark ile Gerçek Zamanlı İşleme

Anomali tespit sonuçları, Apache Kafka ve Spark kullanılarak gerçek zamanlı veri akışı ve analizi için işlenmiştir. Scala uygulaması su bölmeleri içermektedir:

- Tahmin oluşturma için Python betiğiinin çalıştırılması.
 - Apache Spark ile tahmin verisinin yüklenmesi.
 - Anormal ve normal verilerin filtrelenerek ayrıştırılması.
 - Verilerin Kafka başlıklarına gönderilmesi.

1) Veri Ayırıştırma ve Kafka Entegrasyonu: Tahminler şekilde filtrelenmiştir:

- Anomaliler: Tahminler > 0.8 .
 - Normal veriler: Tahminler ≤ 0.2 .

Bu veri akışları, anomalies ve normal_data adlı Kafka başlıklarına gönderilmiştir. Veri akış süreci şekil 63 'de görülmektedir.

Fig. 63: Gerçek zamanlı veri akışı ve işleme boru hattından anomali tespit edilmiş veriler.

Listing 1: Kafka Ortamının Başlatılması

```
# Kafka Ortam n Ba latma
zkserver
kafka-server-start.bat %KAFKA_HOME%\config\server.
    properties

# Kafka Ba l klar n Oluturma
kafka-topics.bat --create --zookeeper localhost:2181 \
    --replication-factor 1 --partitions 1 --topic anomalies
kafka-topics.bat --create --zookeeper localhost:2181 \
    --replication-factor 1 --partitions 1 --topic normal_data

# Kafka Ba l klar ndaki Veriyi Kontrol Etme
kafka-console-consumer.bat --bootstrap-server localhost
    :9092 \
    --topic anomalies --from-beginning
kafka-console-consumer.bat --bootstrap-server localhost
    :9092 \
    --topic normal_data --from-beginning
```

F. Anomali Analizi

Analiz sonucunda, veri setindeki tahmin eşik değeri 0.8'den yüksek olan anomaliler tespit edilmiştir. totalCount adet kayıt işlenirken, anomalyCount adet anomali bulunmuş ve anomali oranı anomalyRatio olarak hesaplanmıştır.

G. Operasyonel Etkiler

Tespit edilen anomaliler, enerji yönetim süreçlerinde potansiyel verimsizliklere işaret etmektedir. Bu verimsizliklerin giderilmesiyle operasyonel iş akışlar optimize edilerek enerji kayıplarının azalması ve maliyet tasarrufu sağlanabilir.

H. Sonuç

Makine öğrenimi ve büyük veri teknolojilerinin entegrasyonu, enerji yönetim süreçlerindeki verimsizlikleri tespit etme ve önleme konusunda önemli bir potansiyel göstermiştir. Gelecekteki çalışmalar, bu çerçeveyin diğer operasyonel alanlara genişletilmesine ve ek tahmin analitiği özelliklerinin entegrasyonuna odaklanacaktır.

VII. SONUÇ VE GELECEK ÇALIŞMALAR (CONCLUSION AND FUTURE WORK)

Bu çalışmada, yüksek raf depolama sistemlerinin enerji yönetimi süreçlerinde verimliliği artırmak ve anomalileri tespit etmek amacıyla makine öğrenimi teknikleri ile büyük veri analitiği araçlarının entegrasyonu sağlanmıştır. Çalışmada önemli sonuçlar elde edilmiş olsa da, bu alandaki sürekli

değişen gereksinimler ve teknolojik gelişmeler ışığında ileriye dönük yapılabilecek çalışmalar şu şekilde belirlenmiştir:

A. Anomali Tespit Algoritmalarının Geliştirilmesi

Makine öğrenimi tabanlı anomali tespit algoritmalarının performansı artırılabilir. Bu bağlamda, özellikle derin öğrenme tabanlı yöntemlerin uygulanması planlanmaktadır. Örneğin, Recurrent Neural Networks (RNN) veya Long Short-Term Memory (LSTM) gibi zaman serisi verisi üzerinde başarılı olan algoritmalar, anomalilerin daha hassas bir şekilde tespiti için kullanılabilir. Ayrıca, mevcut algoritmaların hiperparametre optimizasyonu ve yeni model kombinasyonlarının denemesi hedeflenmektedir.

B. Gerçek Zamanlı Optimizasyon

Apache Kafka'nın daha etkin kullanımı ile gerçek zamanlı veri akışı ve analiz süreçleri iyileştirilecektir. Gerçek zamanlı enerji optimizasyonu sağlayan adaptif mekanizmaların geliştirilmesi hedeflenmektedir. Bu tür bir yaklaşım, sistemin sürekli izlenmesine olanak tanıyarak enerji tüketimindeki dalgalanmaları arasında deneleyebilir ve operasyonel verimliliği artırabilir.

C. Farklı Veri Setlerinin Entegrasyonu

Ceşitli endüstriyel depolama sistemlerinden elde edilen farklı veri setlerinin entegrasyonu ile analiz kapsamı genişletilebilir. Bu sayede, sistemlerin farklı kullanım senaryolarındaki davranışları daha iyi anlaşılabilir. Ayrıca, çoklu veri kaynaklarından elde edilen verilerin ortak bir platformda işlenmesi, çapraz analizlere olanak sağlayarak enerji tüketim dinamiklerini daha geniş bir perspektifte incelemeyi mümkün kılacaktır.

D. IoT Entegrasyonu ve Sensör Ağı Geliştirmeleri

Mevcut sensör altyapısının genişletilmesi ve IoT cihazlarının sistemlere entegre edilmesi ile daha ayrıntılı veri toplanması hedeflenmektedir. IoT tabanlı çözümler sayesinde, depolama sistemlerinin her bir bileşeni daha detaylı şekilde izlenebilir ve enerji tüketimi optimizasyonunda daha yüksek hassasiyet elde edilebilir.

E. Veri Gizliliği ve Güvenliği

Büyük veri analitiği süreçlerinde veri gizliliği ve güvenliği, sürdürülebilir bir sistem geliştirme açısından kritik bir öneme sahiptir. Bu kapsamda, veri şifreleme yöntemlerinin güçlendirilmesi ve erişim kontrol mekanizmalarının geliştirilmesi planlanmaktadır. Özellikle, GDPR gibi veri koruma düzenlemelerine uyum sağlamak için daha sofistike güvenlik protokollerini uygulanacaktır.

F. Kullanıcı Odaklı Raporlama Araçlarının Geliştirilmesi

Sistemden elde edilen analiz sonuçlarının, karar vericilere ve operatörlere daha erişilebilir bir formatta sunulması amacıyla kullanıcı dostu raporlama araçları geliştirilecektir. Bu araçlar, görsel destekli raporlar ve interaktif veri analitiği platformları içerecek şekilde tasarlanacaktır. Böylece, karar alma süreçleri daha hızlı ve etkin hale gelecektir.

G. Yapay Zeka Destekli Karar Destek Sistemleri

Yapay zeka destekli karar destek sistemlerinin geliştirilmesi, yüksek raf depolama sistemlerinin operasyonel süreçlerini optimize etmede önemli bir adım olacaktır. Bu sistemler, enerji tüketimi, operasyonel verimlilik ve anomali yönetimi gibi kritik alanlarda öngörüler sağlayarak karar verme süreçlerini destekleyecektir.

H. Endüstriyel Uygulamalar ve Genelleştirme

Geliştirilen yöntemlerin farklı endüstriyel uygulamalara genelleştirilmesi planlanmaktadır. Yüksek raf depolama sistemlerinin ötesinde, lojistik, üretim ve tedarik zinciri gibi alanlarda benzer yaklaşımlar uygulanabilir. Bu tür uygulamalar, çalışmanın etki alanını genişletecek daha kapsamlı çözümler sunmayı mümkün kılacaktır.

Bu gelecek çalışmalar, enerji verimliliğini artırmaya, operasyonel süreçleri daha sürdürülebilir hale getirmeye ve yüksek raf depolama sistemlerinin genel performansını geliştirmeye yönelik önemli adımlar sağlayacaktır.

REFERENCES

- [1] A. G. Frank, L. S. Dalenogare, and N. F. Ayala, "Industry 4.0 technologies: Implementation patterns in manufacturing companies," *International Journal of Production Economics*, vol. 210, pp. 15–26, 2019. doi: 10.1016/j.ijpe.2019.01.004.
- [2] D. Riordan and M. Jones, "The adoption of Industry 4.0 in manufacturing SMEs," *Procedia Manufacturing*, vol. 38, pp. 415–422, 2019. doi: 10.1016/j.promfg.2019.04.041.
- [3] J. Wan, J. Liu, Z. Shao, X. Zeng, Y. Zhou, and P. Zhang, "Industrial IoT (IIoT) and smart manufacturing: A review," *Advances in Manufacturing*, vol. 7, no. 1, pp. 1–14, 2019. doi: 10.1007/s40436-018-0021-7.
- [4] J. Yoon, H. Choi, and J. Lee, "A survey of anomaly detection in zeki factories," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1229–1240, 2019. doi: 10.1109/TII.2019.2891220.
- [5] T. Chen, H. Xu, H. Liu, and W. Wang, "Zeki üretim için Endüstri 4.0: Gelişmeler ve zorluklar," *Journal of Industrial Integration and Management*, vol. 2, no. 3, pp. 175–192, 2017. doi: 10.1142/S242486221750015X.
- [6] A. Radziwon, A. Bilberg, M. Bogers, and E. Skov Madsen, "The zeki fabrikaları tanımlamak: Perspektifler ve tanımlamalar," *Procedia CIRP*, vol. 17, pp. 74–79, 2014. doi: 10.1016/j.procir.2014.03.113.
- [7] M. Hasan, M. Islam, T. Zarif, and M. Hashem, "Endüstri 4.0'daki güvenlik ve anomali tespit zorlukları," *Procedia Computer Science*, vol. 155, pp. 653–658, 2019. doi: 10.1016/j.procs.2019.08.093.
- [8] H. Hsieh, J. Li, and S. Wang, "Predictive maintenance in smart factories: A comprehensive review," *IEEE Access*, vol. 7, pp. 100437–100447, 2019. doi: 10.1109/ACCESS.2019.2928995.
- [9] A. Bagozi, M. Trevisan, and E. Veneti, "Real-time anomaly detection for zeki factories," *Journal of Manufacturing Systems*, vol. 45, pp. 305–315, 2017. doi: 10.1016/j.jmsy.2017.06.005.