Kumaş Veri Tabanında Kusur Tespit Yöntemleri ve Sonuçları

Özet:

Kusurların tespiti ve sınıflandırılması için görüntü işlemenin bilim ve endüstride uzun bir süredir kullanıldığı bir gerçektir. Bu süreçte, sürekli olarak yeni yöntemler geliştirilmekte ve her yönüyle iyileştirmeler sağlanmaktadır. Kumaş kusurlarının bölümlendirilmesi, tekstil kalite kontrol süreçlerinin ayrılmaz bir unsuru olarak kabul edilir. Ancak, yüksek kaliteli açıklamalı verilerin sınırlı olması ve kumaş kusurlarının geniş bir çeşitliliği, bu alanda yapay zekâ uygulamalarını zorlaştıran önemli zorluklar oluşturmaktadır. Bu faktörler, mevcut modellerin genelleme ve bölümlendirme performansını kısıtlamakta, çeşitli kumaş türleri ve kusur çeşitliliği karşısında etkili bir şekilde başa çıkma yeteneklerini sınırlamaktadır.

Bu makalenin temel amacı, tespit ve sınıflandırma yöntemlerini doğrudan karşılaştırabilme imkânı sunmak için kamuya açık bir şekilde açıklamalı bir kıyaslama yapmaktır. Bu bağlamda, kusurlu ve kusursuz örnekleri içeren geniş bir kumaş görüntü seti kullanılacaktır. Ayrıca, farklı yöntemler detaylı bir şekilde incelenmiş ve bu yöntemlerden beşi seçilerek, belirlenen görüntü kümesine uygulanmıştır. Elde edilen sonuçlar da bu makalede detaylı bir şekilde sunulmaktadır.

Anahtar Kelimeler: Kumaş kusurları, kusur sınıflandırması, segmentasyon

Methods and Results of Defect Detection in Fabric Databases

Abstract:

It is a well-known fact that image processing has been used in science and industry for a long time for the detection and classification of defects. Throughout this process, new methods are constantly being developed and improvements are made in every aspect. Segmentation of fabric defects is considered an integral part of textile quality control processes. However, the limited availability of high-quality annotated data and the wide variety of fabric defects pose significant challenges for artificial intelligence applications in this field. These factors limit the generalization and segmentation performance of existing models, restricting their ability to effectively handle various types of fabrics and defect diversity.

The primary aim of this article is to provide a publicly available, annotated comparison to directly compare detection and classification methods. In this context, a wide fabric image dataset containing both defective and non-defective samples will be used. Additionally, different methods have been thoroughly examined, and five of these methods have been selected and applied to the designated image set. The obtained results are also presented in detail in this article.

Keywords: Fabric defects, defect classification, segmentation

1. Giriş

Tekstil endüstrisindeki denetim süreçlerinde otomasyon, 1990'lı yılların ortalarından bu yana incelenen bir alandır. Bu ihtiyaç genel olarak endüstride ve özel olarak tekstil endüstrisinde [1, 2] geniş çapta incelenmiştir. Kumaş kusur segmentasyonu tekstillerin kalite kontrolünde zaman açısından maliyetlerin azaltılması ve dolayısıyla müşteri memnuniyeti açısından çok önemli bir rol oynar [3]. Delik, leke, iplik kopması gibi kumaş kusurları kumaş kalitesini etkileyen başlıca faktörlerdir. Bu nedenle kumaşlardaki kusurların etkili ve doğru bir şekilde tespit edilmesi ve segmentlere ayrılması, tekstil kalitesinin sağlanması, üretim verimliliğinin artırılması ve üretim maliyetlerinin azaltılması açısından büyük önem taşımaktadır [4]. Özellikle modern tekstil endüstrisinde, üretim ölçeğinin genişlemesi ve üretimin artmasıyla birlikte Kumaş kusurlarının manüel olarak tanımlanmasına ve bölümlendirilmesine dayanan verimliliğin arttırılması artık üretim ihtiyaçlarını karşılayamamaktadır [5]. Bu nedenle, verimli ve doğru kumaş kusur bölümlendirme teknolojilerinin araştırılması ve geliştirilmesi, tekstil endüstrisi alanında çözülmesi gereken acil sorunlar haline gelmiştir [2]. Ancak bu basit bir iş değildir ve tekstil hata tespitindeki zorluklar göz önüne alındığında son 20 yılda çeşitli yöntemler sunulmuştur [6].

Farklı yazarların sonuçlarının karşılaştırılması, kullanılan görüntülere ilişkin bilgi eksikliği, kusur türlerinin oldukça farklı olması ve çözünürlüğün aynı olmaması nedeniyle imkânsız olmasa da zordur. Ek olarak, [7] ve [8] numaralı kaynaklarda belirtildiği gibi, çok sayıda kumaş hatası tespiti algoritması ve tekniği nedeniyle, kumaş hatası tespit yöntemleri arasında etkili bir karşılaştırma son derece önemli olacaktır, ancak çoğu çalışma farklı veri tabanları, farklı görüntüleme sistemleri ve farklı parametreler kullanmaktadır. Bu, kamuya açık, açıklamalı uygun bir kıyaslama noktasının bulunmamasının, araştırmacıların kendi algoritmalarının mevcut algoritmalara göre avantajlarını niceliksel olarak değerlendirmelerini zorlaştırdığı ve dolayısıyla bu eksikliğin

bazı alanlarda etkili ve uygulanabilir algoritmaların geliştirilmesinde ciddi bir sınırlama oluşturduğu anlamına gelir. Örneğin, şerit içi algılama [9] ve bu aynı zamanda tekstil endüstrisinde de gerçekleşmektedir bu nedenle, farklı yazarların çalışmaları ve karşılaştırmaları için kullanılabilecek halka açık bir veri tabanına sahip olmak önemlidir, çünkü birçok durumda, [7] numaralı kaynakta belirtildiği gibi, çalışmaların çoğunda "yazarlar görüntüleri fabrika ortamlarından alarak veya laboratuvara getirerek kendi veri tabanlarını oluştururlar ve veri tabanı farklı aydınlatma ayarlarıyla oluşturulur. Dolayısıyla yöntemlerin güvenirliği ve geçerliliği objektiflikten uzaktır."

Bu makalenin amacı, kusurlu ve hatasız kumaşların kamuya açık bir veri tabanı kullanarak farklı algoritmaların sonuçlarını karşılaştırmak, böylece şu anda mevcut olan farklı yöntemlerin ve gelecekteki önerilerin doğru bir şekilde karşılaştırılabilmesi mümkün olur. Böylece makalede bahsedilen her yöntemin avantajları bu veri tabanı kullanılarak yeterince değerlendirilebilir. Bir sonraki bölümde kumaşlarda kusur tespiti konusunda yapılan farklı çalışmalar incelenmektedir. Geri kalan bölümlerde uygulanan yöntemlerin bir incelemesi sunulmaktadır. Ayrıca veri tabanı olarak "AITEX Fabric Image Database" [10] kullanılmıştır ve son olarak uygulanan yöntemlerin sonuçları verilmiştir. Bu verilerle araştırmacılar yeni önerilen yöntemleri kolaylıkla karşılaştırabilecekler.

2. Önceki Çalışmalar

Önceki çalışmalar piksel bazında etiketlenmiş verilerin mevcudiyetine bağlı olarak iki türe ayrılabilir. İlk kategori, model eğitimi için piksel bazında etiketlenmiş veriler gerektiren denetimli öğrenme yöntemlerini içerir. İkinci kategori, piksel bazında etiketlenmiş verileri kullanmayan veya eğitim için yalnızca hatasız örnekleri kullanan yöntemleri içerir. Bu, zayıf denetimli öğrenme yönteminin bir türü olarak düşünülebilir.

Birinci tür çalışmada, piksel bazında eğitim verilerinin kullanılması nedeniyle, birçok çalışma, karşılık gelen veri kümeleri üzerinde kayda değer segmentasyon performansı elde etmiştir. Örneğin, Cheng ve ekibi [11], Ayrı Evrişimli UNet'ye (SCUNet) dayanan bir kumaş hatası tespit yöntemi önerdiler ve AITEX veri kümesinde %98,01 doğruluk elde ettiler. Huang ve ekibi [12], kumaş kusur tespiti için, 50 kadar az kusur numunesi ve gerçek zamanlı tespit hızları ile yüksek doğrulukta kusur lokalizasyonu elde eden, sekiz ileri teknoloji yöntem arası doğruluk ve doğruluktan daha iyi performans gösteren, oldukça verimli bir sinir ağı önerdiler. Liu ve ekibi [13], kumaş kusur tespiti için bir dikkat mekanizmasına sahip, tespit performansını başarılı bir şekilde artıran ve hesaplama taleplerini önemli ölçüde artırmadan kusur lokalizasyonunda en gelişmiş yöntemleri geride bırakan derin bir belirginlik modeli önerdi. Kumaş kusur tespiti için derin öğrenmeye ve denetimli öğrenmeye dayalı benzer çalışmalar arasında [14] – [20] yer almaktadır.

Ancak denetimli öğrenme büyük ölçüde büyük ölçekli ve yüksek kaliteli etiketli verilere dayanır. Kumaş kusuru görüntülerini toplamak ve açıklama eklemek zor olabilir, bu da kumaş kusuru segmentasyonu için yalnızca denetimli öğrenmeye güvenmeyi pratik hale getirmez. Bazı araştırmacılar, kusur bölümleme için daha fazla hatasız görüntü bilgisi ve diğer açıklama bilgisi türlerini kullanarak piksel bazında açıklama edinme sorununu çözmeye çalışmışlardır. Örneğin, Koulali ve ekibi [21] eğitim için yalnızca hatasız bir numuneye ihtiyaç duyan bir yöntem önerdiler ve Desenli Kumaşlar kıyaslama veri kümesinde rekabetçi performans sergilediler. Liu ve ekibi [22], çeşitli hata türleri için derin bir semantik segmentasyon ağını özelleştiren ve yeni hatasız numuneler üzerinde makul kusurlar oluşturmak için çok aşamalı bir GAN kullanan, kumaş kusur tespiti için GAN tabanlı bir çerçeve önerdi. Yaklaşımları sürekli veri kümesi güncellemelerine olanak tanır ve değişen koşullar altında kusur tespit performansını artırmak için ağın ince ayarına katkıda bulunur. Bununla birlikte, bu yöntemler, tamamen denetlenen yöntemlere kıyasla kusurların kesin konumlarını ve şekillerini elde etmedeki zorluktan dolayı genellikle kusur bölümlemede daha düşük doğrulukla mücadele etmektedir. Zayıf denetim, sınırlı piksel düzeyindeki açıklama verilerine dayandığından, daha fazla yineleme ve karmaşık optimizasyon algoritmaları gerektirdiğinden, model eğitimi de zorlayıcı olabilir. Farklı veri kümelerindeki performans kararsız olabilir, bazen denetlenen yöntemlerle eşleşebilir veya onları aşabilir, diğer durumlarda ise düşük performans gösterebilir [23], [24].

3. Materyal ve Yöntem

3.1 Veri Tabanı

Bu çalışmada AITEX [10] isimli kumaş hatası tespit veri seti kullanılmıştır. AITEX veri kümesi kendi kendine toplanan ve kamuya açık bir veri kümesidir. AITEX veri kümesi, orijinal görüntülerin boyutlarına tam olarak karşılık gelen piksel düzeyinde açıklamalar içerir.

Kusurların tespiti ve sınıflandırılması alanında araştırma çalışmasını yürütmek için (Tablo 1), kullanılan her yöntemin sonuçlarının geliştirilmesine ve değerlendirilmesine olanak tanıyan, kusurlu ve kusursuz temsili bir numune koleksiyonunun mevcut olması önemlidir. Tablo 1, bazı kumaş kusur türlerini özetlemektedir ve bunlar kaynak [25-27]'de açıklanmıştır. Veri tabanında mevcut olan kusurlar (*) ile işaretlenmiştir. 12 kusurun 61'e kıyasla az görünebileceğine dikkat edilmelidir, ancak değerlendirilen önceki çalışmalar dikkate alındığında en fazla sayıda kusur içeren ve kamuya açık olan veri tabanıdır. Daha genel yönleri analiz etmek için araştırmacılar tarafından kullanılan birçok farklı görüntü veri tabanı türü vardır.

Tablo 1. Kumaş kusur türleri

Defect type	Defect number	Defect type	Defect number	Defect type	Defect number
Floats	1	Cut selvage*	22	Net multiples	43
Broken end*	2	Crease*	23	Loom fly	44
Oil stains	3	Warp float	24	Missing draw	45
Slubs	4	Warp Ball*	25	Missing weft	46
Miss end	5	Foreign fiber	26	Kink	47
Broken yarn*	6	Knots*	27	Unrelated corpus	48
Miss pick	7	Harness breakdown	28	Burl	49
Spot	8	Contamination*	29	Colorfly	50
Big knot	9	Nep*	30	Broken needle	51
Broken pick*	10	Water damage	31	Barre	52
End out	11	Thick bar	32	Dropped stitch	53
Lines	12	Coarse end	33	Warp lacking	54
Fault yarns	13	Coarse filling	34	Open reed	55
Wrong draw	14	Knees	35	Soiled end	56
Dirty yarn	15	Weft crack*	36	Sloughed filling	57
Weft curling*	16	Ripped	37	Gout	58
Double weft	17	Double yarn	38	Knot with halos	59
Trip warp	18	Miss yarn	39	Thick node	60
Fuzzy ball*	19	Broken fabric	40	Holes	61
Slack end	20	Roving	41		
Thin bar	21	Thin place	42	No defect	00

*Veri tabanımızda mevcut olan kusurlar

Tekstil kumaş veri tabanı 7 farklı kumaşa ait 245 görselden oluşmaktadır. Her kumaş türü için 20 adet olmak üzere 140 adet hatasız görüntü bulunmaktadır. Farklı kusur türlerine sahip 105 adet görüntü bulunmaktadır. Görsellerin boyutu 4096×256 pikseldir [10].

3.2 Verileri Ön İşleme

Makine öğrenimi modelleri eğitmek için kullanılan veri setlerini hazırlamak, genellikle bir dizi ön işleme adımını içerir. İlk olarak, orijinal veri setini kusurlu ve kusursuz görüntüler için ayrı ayrı klasörlere bölmek bu adımlardan biridir. Bu görevi gerçekleştirmek için, "split_dataset()" fonksiyonu kullanılır. Bu fonksiyon, "Defect_images" ve "NODefect_images" adlı iki çıkış klasörü oluşturur. Kusurlu görüntüler, kaynak klasörden "Defect_images" klasörüne kopyalanırken, kusursuz görüntüler ilgili alt klasörlerden "NODefect_images" klasörüne kopyalanır.

Ardından, veri setindeki kusurlu sınıfın örnek sayısını artırmak için veri artırma teknikleri uygulanır. Keras'ın ImageDataGenerator'ı, rastgele dönme, kaydırma, yaklaştırma, çevirme ve parlaklık ayarı gibi rastgele dönüşümleri uygulayarak yeni görüntüler oluşturmak için kullanılır. Bu, modelin daha iyi genelleme yapmasına ve öğrenmesine yardımcı olabilir. Veri artırma, bir makine öğrenimi modelini eğitirken, modelleri mevcut verilerin biraz değiştirilmiş birkaç kopyası üzerinde eğiterek fazla uydurmayı(overfitting) azaltmak için kullanılan bir makine öğrenimi tekniğidir.[28]

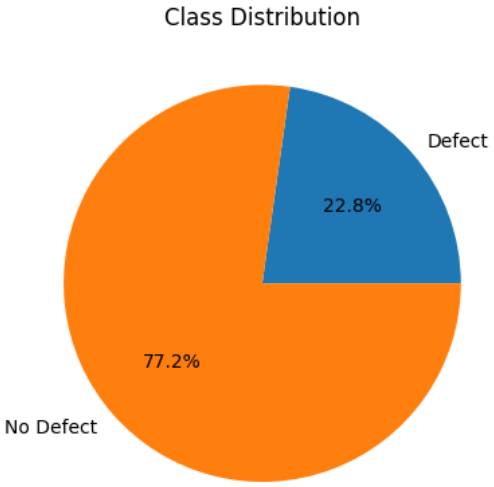
Artırılmış kusurlu görüntüler ve kusursuz görüntüler, tek bir veri setinde birleştirilir. Bu, bir özellik matrisi (X) ve bir etiket dizisi (y) oluşturur. X, artırılmış kusurlu görüntüleri ve kusursuz görüntüleri içerirken, y, kusurlu ve kusursuz sınıfları temsil eden etiketleri içerir.

Rastgele sıralama, öğrenme algoritmalarının modelin bir sınıftan diğerine geçişini daha etkili bir şekilde öğrenmesine yardımcı olabilir. Bu nedenle, scikit-learn'ün utils modülünden "shuffle()" fonksiyonu kullanılarak veri seti karıştırılır. Daha sonra, veri seti, modelin beklediği 4D tensör formatına geri dönüştürülür (örnek sayısı, görüntü yüksekliği, görüntü genişliği, kanal sayısı).

Veriyi Eğitim ve Test Setlerine Ayırma: Veri seti, eğitim ve test setleri olarak ayrılır. Bu işlem, scikit-learn'ün "train_test_split()" fonksiyonu kullanılarak gerçekleştirilir. Belirtilen test boyutu, verinin %20'sinin test için kullanılacağını ve geri kalan %80'inin eğitim için kullanılacağını belirler.

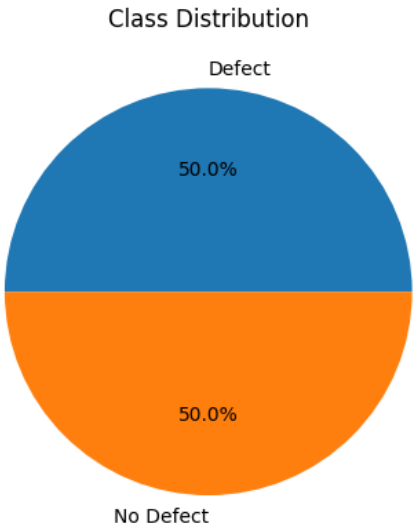
Sınıf dağılımı bir veri setinde bulunan farklı sınıfların (etiketlerin) sayısal dağılımını ifade eder. Bir makine öğrenimi veya veri madenciliği problemi genellikle örneklerin belirli sınıflara ait olup olmadığını belirlemeye çalışır. Örneğin, bir görüntü sınıflandırma görevinde, sınıflar, görüntülerin içerdikleri nesnelere veya kavramlara göre belirlenebilir.

Sınıf dağılımı, bir veri setindeki her bir sınıfın ne kadar sıklıkta temsil edildiğini ifade eder. Eğer bir veri setindeki sınıflar arasında eşit bir dağılım varsa, yani her sınıfın yaklaşık olarak aynı sayıda örnek içerdiği bir durum söz konusuysa, bu durumda sınıf dağılımı dengeli olarak adlandırılır. Ancak, eğer bir veya birkaç sınıf diğerlerine göre belirgin şekilde daha fazla örnek içeriyorsa, bu durumda sınıf dağılımı dengesizdir. Dengesiz sınıf dağılımı, özellikle makine öğrenimi modelleriyle çalışırken önemli bir durumdur, çünkü modelin eğitimini etkileyebilir. Dengesiz sınıf dağılımına sahip bir veri seti, modele, çoğunluk sınıfındaki örnekleri öğrenmeye daha fazla odaklanma eğiliminde olabilir, bu da modelin azınlık sınıfındaki örnekleri doğru bir şekilde öğrenmekte zorlanmasına neden olabilir.



Şekil 1. Sınıf Dağılımı

Bu nedenle, sınıf dağılımının dengeli olup olmadığına dikkat etmek ve gerektiğinde dengesizliği ele almak, modelin iyi performans göstermesi açısından önemlidir. Sınıf dengesizliğini gidermek için Sentetik Azınlık Üzerine Örnekleme Tekniği (SMOTE) kullanılır [29]. Bu, azınlık sınıfına ait veri sayısını artırarak sınıf dengesizliğini düzeltmeye yardımcı olur. Ancak, bu adım yalnızca eğitim verilerine uygulanmalıdır, böylece test verilerinden bilgi sızmasını önleriz.



Şekil 2. SMOTE uygulandıktan sonraki dağılım

Sonrasında görüntülere renk uzayı dönüşümü uygulayarak gri tonlamayla renk özelliklerini incelemek verimizi anlamak açısından büyük fayda sağlayacaktır ama bizim veri setimiz zaten gri tonlamaya dönüştürülmüş bir veri setidir. Gri tonlamaya dönüştürülmüş bir veri seti olduğu için kenar algılama çeşitlerinden biri olan Canny yöntemini kullanmaya karar verdim.

Kenar algılama, görüntü işlemede yaygın olarak kullanılan bir tekniktir ve bir görüntünün içindeki anlamlı sınırları veya kenarları belirlemeyi amaçlar. Bu işlem, genellikle görüntüdeki renk veya yoğunlukta ani değişiklikleri tespit ederek yapılır. Kenar algılama, nesne tespiti, görüntü segmentasyonu ve daha birçok görüntü işleme uygulamasında önemli bir rol oynar.

Canny, en popüler kenar algılama yöntemlerinden biridir ve birden fazla adımdan oluşur. Bu adımlar gradyanın hesaplanması, maksimum gradyanın tespiti, ve histeresis eşikleme içerir. Canny algoritması, güçlü ve zayıf kenarları ayırt edebilir ve yanlış pozitifleri azaltmaya yardımcı olacak şekilde tasarlanmıştır.

3.3 Kumaş Kusur Sınıflandırılmasında CNN Algoritmasının Kullanılması

CNN'ler, özellikle görüntü işleme ve sınıflandırma görevlerinde sıkça kullanılan derin öğrenme yapılarıdır. CNN'ler, görüntülerin yüksek seviyedeki karmaşık özelliklerini otomatik olarak öğrenmek için tasarlanmıştır ve genellikle şu bileşenleri içerir:

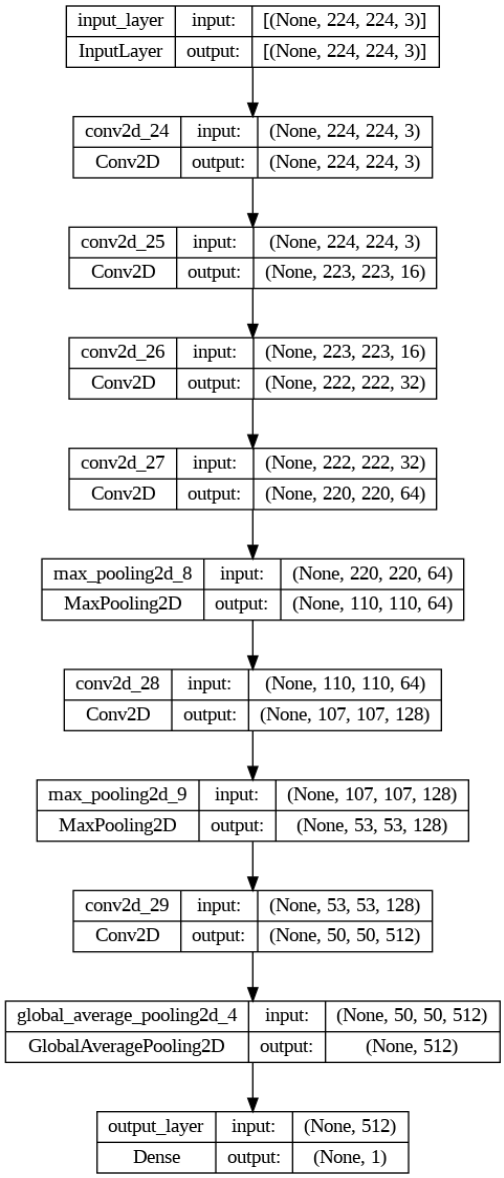
Evrişim Katmanları (Convolutional Layers): Görüntü üzerinde yerel özellikleri (kenarlar, doku vb.) tespit etmek için kullanılır. Evrişim işlemi, küçük bir pencere (kernel) kullanılarak tüm görüntü üzerinde kaydırılarak gerçekleştirilir.

Aktivasyon Fonksiyonları: Genellikle ReLU (Rectified Linear Unit) kullanılır. Non-linearity (doğrusal olmayanlık) ekleyerek modelin karmaşık örüntüleri öğrenmesine yardımcı olur.

Havuzlama Katmanları (Pooling Layers): Görüntünün boyutunu azaltırken önemli özellikleri korur. Max pooling en yaygın kullanılan yöntemdir.

Tam Bağlantılı Katmanlar (Fully Connected Layers): Evrişim ve havuzlama işlemlerinden sonra elde edilen özellikleri kullanarak sınıflandırma yapar.

Model 1: CNN Modeli



Yukarıdaki model, CNN modelinin katman yapılandırmasını ve her katman arasındaki tensör boyutlarının nasıl değiştiğini gösteren bir model özetdir. Modelimizin her katmanındaki girdi ve çıktı boyutlarını açıklayalım:

1. **InputLayer**: Modelimize girdi olarak verilen görüntülerin boyutu (224, 224, 3) olarak belirtilmiştir. Bu, görüntülerin 224x224 piksel boyutunda ve 3 renk kanalına (genellikle RGB) sahip olduğunu gösterir.
2. **conv2d_24 (Conv2D)**: İlk evrişim katmanıdır ve görüntünün boyutunu değiştirmeden (224, 224, 3) çıktı üretir. Bu, padding'in 'same' olarak ayarlandığını gösterir.
3. **conv2d_25 (Conv2D)**: Bu katman daha fazla özellik haritası üretir (16 adet) ve görüntü boyutunu bir miktar azaltır (223, 223). Bu azalma, padding olmadığı veya 'valid' olduğu anlamına gelir.
4. **conv2d_26 (Conv2D)**: Daha fazla özellik haritası (32 adet) çıkarır ve görüntü boyutunu (222, 222) azaltır.
5. **conv2d_27 (Conv2D)**: 64 özellik haritası çıkarır ve boyutu daha da (220, 220) düşürür.
6. **max_pooling2d_8 (MaxPooling2D)**: Maksimum havuzlama katmanı, boyutu yarı yarıya azaltır (110, 110) ve özellik haritası sayısını değiştirmez (64).
7. **conv2d_28 (Conv2D)**: 128 özellik haritası çıkarır ve boyutu (107, 107) azaltır.
8. **max_pooling2d_9 (MaxPooling2D)**: İkinci maksimum havuzlama katmanı, boyutu yine yarı yarıya azaltır (53, 53) ve özellik haritası sayısını değiştirmez (128).
9. **conv2d_29 (Conv2D)**: Çok daha fazla özellik haritası (512 adet) çıkarır ve boyutu (50, 50) azaltır.
10. **global_average_pooling2d_4 (GlobalAveragePooling2D)**: Her özellik haritası için ortalama havuzlama yaparak sabit boyutlu bir vektör üretir (512). Bu, her bir özellik haritasından tek bir ortalama değer alır.
11. **output_layer (Dense)**: Tam bağlı katman veya yoğun katman olarak da bilinir, modelin son çıktısını üretir. Bu örnek için çıktı boyutu (1), modelin muhtemelen ikili bir sınıflandırma görevi için eğitildiğini gösterir.

Bu özet, modelimizin katman katman nasıl bir işlem yaptığını ve her bir katmanın tensör boyutlarını nasıl değiştirdiğini gösterir. Modelin son katmanındaki tek nöron çıktısı, ikili bir sınıflandırma (var/yok, evet/hayır, pozitif/negatif gibi) için kullanıldığını göstermektedir. Bu tür bir model, görüntüden karmaşık özellikleri otomatik olarak çıkarmak ve bu özellikler üzerinden sınıflandırma yapmak için kullanılır.

Tablo 2. CNN Metrik Sonuçları

	Precision	Recall	F1-score	Support
0.0	0.79	1.00	0.89	27
1.0	1.00	0.94	0.97	108
macro avg	0.90	0.97	0.93	135
weighted avg	0.96	0.95	0.95	135
accuracy			0.95	135

Tabloyu incelediğimizde, CNN modelimizin oldukça iyi performans gösterdiğini söyleyebiliriz.

1. **Sınıf 0.0**:
    - **Precision (Kesinlik)**: 0.79. Bu değer, modelin %79 oranında, kusursuz olarak işaretlediği örneklerin gerçekten kusursuz olduğunu gösterir.
    - **Recall (Duyarlılık)**: 1.00. Modelin, gerçekte kusursuz olan tüm örnekleri %100 oranında doğru bir şekilde tespit ettiğini belirtir.

- **F1-Score**: 0.89. Kesinlik ve duyarlılığın harmonik ortalaması olan F1 skoru, her iki metriği dengeli bir şekilde birleştiren bir değerdir. Bu durumda, %89 oldukça yüksek bir değerdir ve modelin kusursuz örnekleri iyi tanıdığını gösterir.

2. **Sınıf 1.0**:

   - **Precision**: 1.00. Modelin kusurlu olarak işaretlediği tüm örneklerin gerçekten kusurlu olduğunu gösterir. Bu, yanlış pozitiflerin (yanlışlıkla kusurlu olarak işaretlenen kusursuz örnekler) olmadığını gösterir.
   - **Recall**: 0.94. Gerçekte kusurlu olan örneklerin %94'ünü doğru bir şekilde tespit ediyor.
   - **F1-Score**: 0.97. Kusurlu örnekler için F1 skoru, modelin bu sınıf üzerinde mükemmel bir dengeli performans gösterdiğini gösterir.

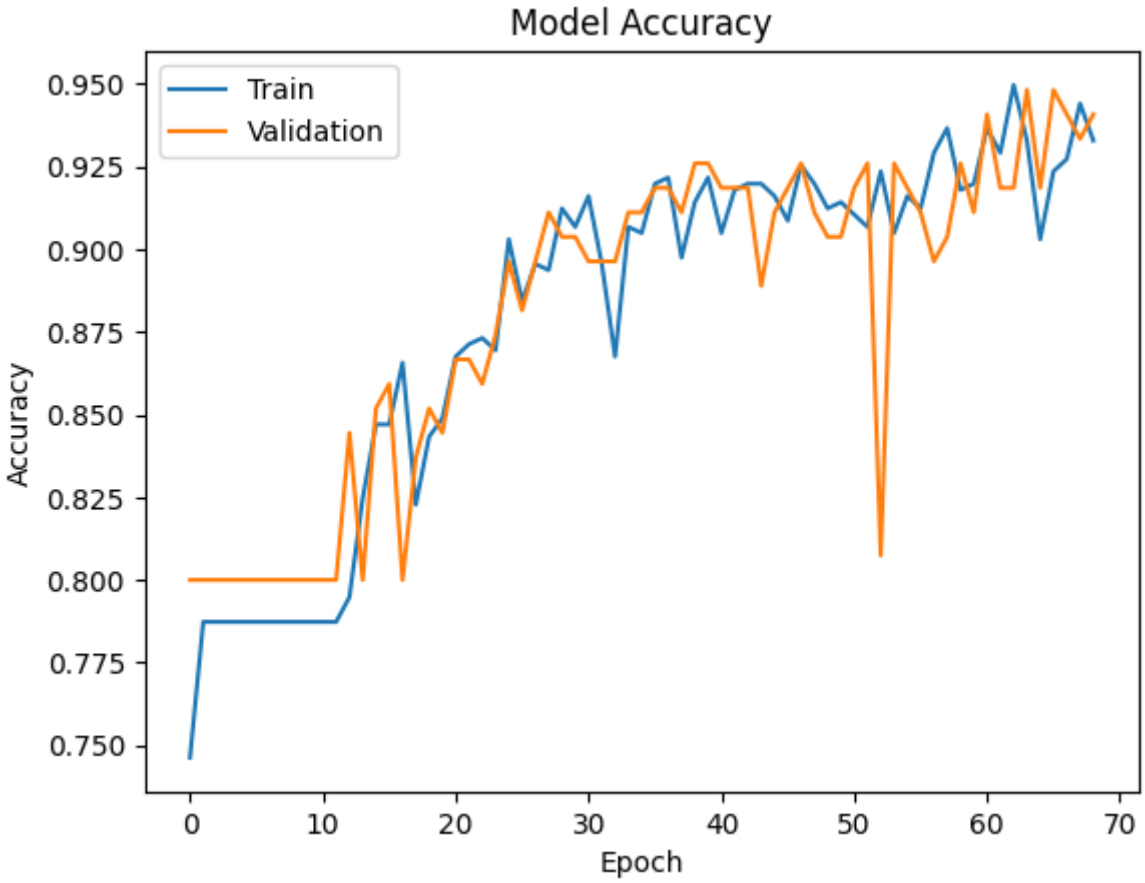
3. **Genel Değerlendirmeler (macro ve weighted averages)**:

   - **Macro Avg**: Her sınıfın eşit derecede önemli olduğu durumlarda kullanılır. Macro avg değerleri, modelin her iki sınıfı da benzer bir performansla tanıdığını gösterir.
   - **Weighted Avg**: Sınıfların desteklerine göre ağırlıklandırılmış ortalamaları temsil eder. Bu, modelin sınıflandırma performansının destek sayısıyla orantılı olarak ağırlıklandığını gösterir. Bu durumda, weighted average'ler de oldukça yüksek, bu da modelin genel olarak iyi performans gösterdiğini belirtir.

4. **Genel Accuracy (Doğruluk)**: %95. Bu oran, modelin genel olarak test setindeki tüm örneklerin %95'ini doğru bir şekilde sınıflandırdığını gösterir.

Bu değerler, modelimizin genel olarak mükemmel performans gösterdiğini ve her iki sınıfı da yüksek doğrulukla ayırt edebildiğini göstermektedir.

Grafik 1. CNN Accuracy Grafiği



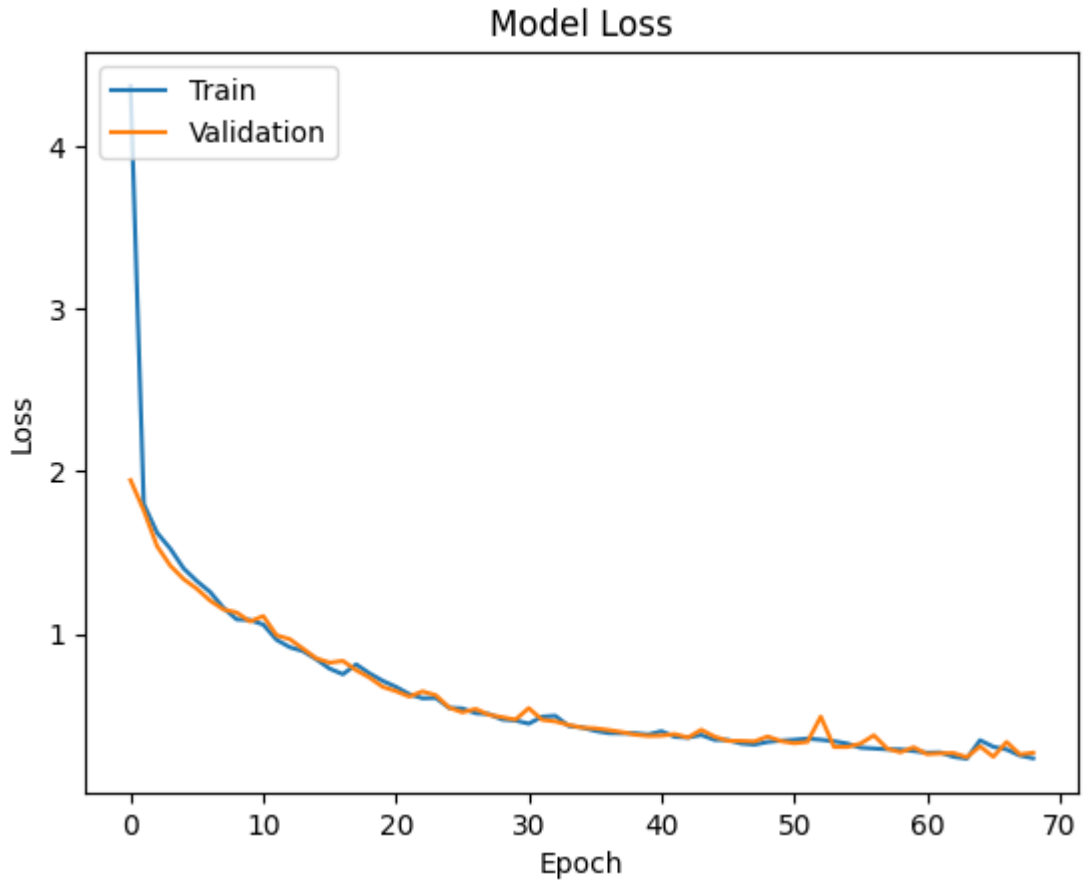
Doğruluk(accuracy) grafiğine göre, CNN modelimizin eğitim süreci boyunca şu gözlemleri yapabiliriz:

1. **Eğitim Doğruluğu (Mavi Çizgi)**: Model eğitim seti üzerinde zamanla giderek daha yüksek doğruluk elde etmiş. Grafiğin sonlarına doğru eğitim doğruluğu yükselmeye devam ediyor ve %95 civarında stabilize olmuş görünüyor. Bu, modelin eğitim verilerini öğrenme kabiliyetinin yüksek olduğunu gösterir.

2. **Doğrulama Doğruluğu (Turuncu Çizgi)**: Doğrulama doğruluğu da genel olarak artmış, ancak eğitim doğruluğundan biraz daha düşük ve daha fazla dalgalanma göstermiş. Bu, modelin eğitim verilerine kıyasla doğrulama verilerinde bir miktar aşırı uyuma (overfitting) eğiliminde olabileceğine işaret edebilir.

3. **Doğruluk Farkı**: Eğitim ve doğrulama doğrulukları arasında sürekli bir fark var, ama bu fark çok büyük değil. Eğer bu fark büyük olsaydı, aşırı uyum daha belirgin bir problem olabilirdi.

Grafik 2. CNN Loss Grafiği



Yukarıdaki kayıp (loss) grafiği, CNN modelinin eğitim sürecinin bir parçası olarak kayıp değerlerinin nasıl değiştiğini gösteriyor. Grafikteki eğriler, eğitim ve doğrulama setleri üzerindeki kaybı epoch sayısına göre göstermektedir. Grafiğin açıklaması ve yorumu:

1. **Eğitim Kaybı (Mavi Çizgi)**: İlk epoch'ta eğitim kaybı hızla düşmüş. Bu, modelin hızlı bir şekilde öğrenmeye başladığını ve büyük ölçüde iyileştiğini gösterir. Epochlar ilerledikçe kayıp değeri daha da azalmış ve belirli bir seviyede istikrar kazanmış.

2. **Doğrulama Kaybı (Turuncu Çizgi)**: Doğrulama kaybı, eğitim kaybıyla benzer bir trend izlemiş. İlk başta hızla düşmüş ve daha sonra epochlar ilerledikçe azalmış ve eğitim kaybına yakın bir değerde istikrar kazanmış.

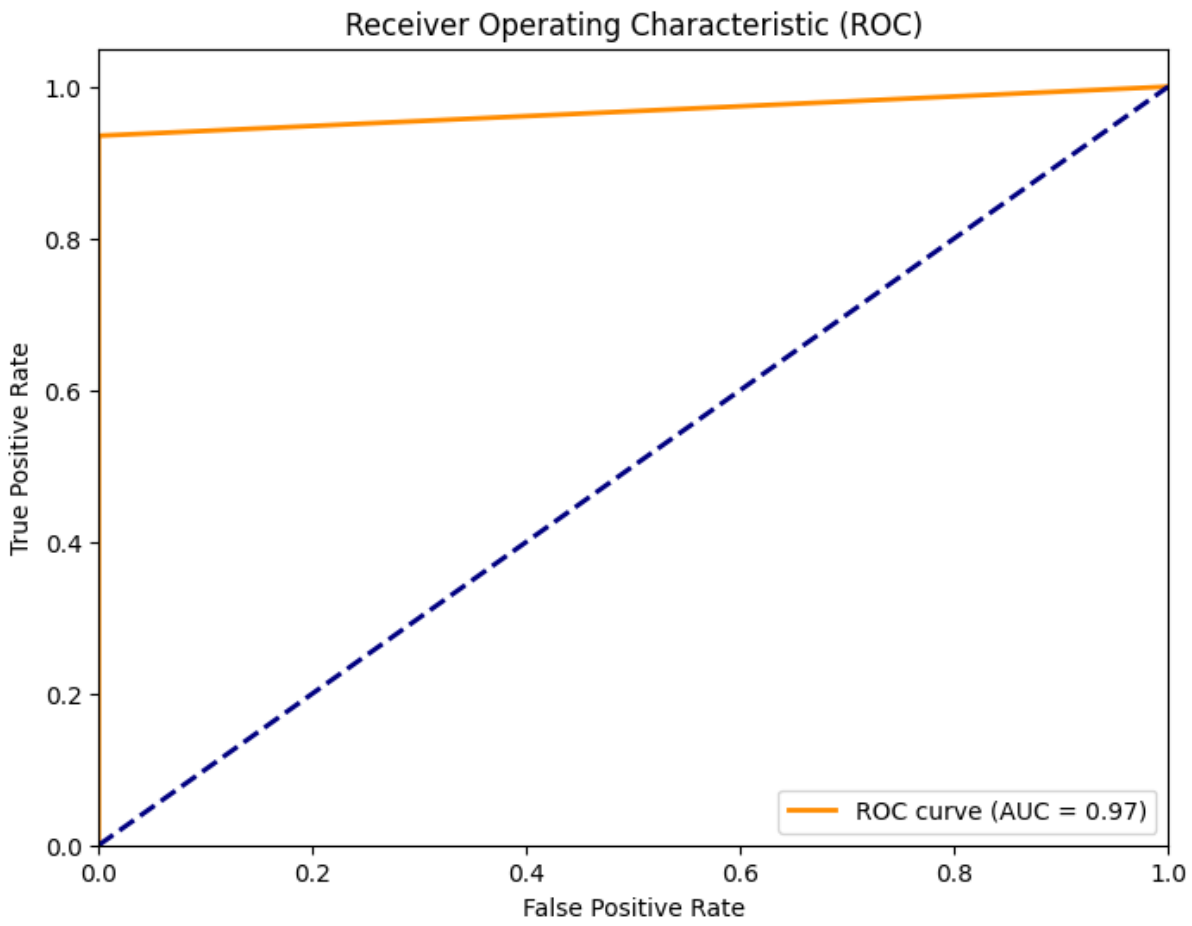
3. **Eğitim ve Doğrulama Kaybı Arasındaki İlişki**: Eğitim ve doğrulama kaybının yakın ve benzer bir trendde olması, modelimizin aşırı uyuma (overfitting) belirtileri göstermediğini gösterir. Eğer

doğrulama kaybı, eğitim kaybından önemli ölçüde yüksek olmuş olsaydı, bu aşırı uyumun bir işareti olabilirdi.

4. **Düşük ve İstikrarlı Kayıp Değerleri**: Düşük ve istikrarlı bir kayıp değeri, modelin verilerdeki örüntüleri başarıyla öğrendiğini ve bu öğrenmeyi doğrulama setine genelleştirebildiğini gösterir.

Grafikteki dalgalanmalar, özellikle doğrulama seti üzerinde, modelin farklı veri noktalarında performansının değişkenlik gösterebileceğini işaret eder. Ancak bu dalgalanmalar, eğitim sürecinin doğal bir parçası olabilir ve modelin genel performansını negatif bir şekilde etkilemiyor.

Grafik 3. CNN ROC Grafiği



Yukarıdaki ROC (Receiver Operating Characteristic) eğrisi, CNN modelinin sınıflandırma performansını değerlendirmek için kullanılan bir araçtır. ROC eğrisi, farklı eşik değerleri üzerinde modelin doğru pozitif oranı (True Positive Rate, TPR) ile yanlış pozitif oranı (False Positive Rate, FPR) arasındaki ilişkiyi gösterir.

TPR, pozitif olarak sınıflandırılması gereken örneklerin yüzdesidir ve FPR, yanlışlıkla pozitif olarak sınıflandırılan negatif örneklerin yüzdesidir.

ROC eğrisi ve AUC (Area Under the Curve) değeri hakkında şunları söyleyebiliriz:

- **AUC Değeri**: AUC değeri 0.97 olarak belirtilmiş, bu çok yüksek bir değerdir ve mükemmel sınıflandırma performansını gösterir. AUC değeri 1'e ne kadar yakınsa, model o kadar iyi sınıflandırma yapabilir demektir. 0.97 değeri, modelin pozitif ve negatif sınıfları ayırt etme konusunda çok başarılı olduğunu gösterir.

- **Eğri Şekli**: ROC eğrisi sol üst köşeye çok yakın. Bu, modelin düşük yanlış pozitif oranı ile yüksek doğru pozitif oranı elde ettiğini, yani az sayıda yanlış ile birçok gerçek pozitifi doğru bir şekilde tespit ettiğini gösterir.

- **Karar Eşiği**: ROC eğrisi, modelin farklı karar eşiklerindeki performansını gösterir. Bu modelde, karar eşiğini değiştirerek modelin duyarlılığını veya özgüllüğünü ayarlayabilirsiniz. Yüksek bir eşik, modelin

daha az yanlış pozitif üretmesine neden olabilir, ancak bazı doğru pozitifleri kaçırabilir. Daha düşük bir eşik, daha fazla doğru pozitifi yakalamanızı sağlar, ancak yanlış pozitif sayısını da artırabilir.

Sonuç olarak, modelimizin test setinde gösterdiği performans, ROC eğrisine göre oldukça yüksek ve güvenilir.

3.4 Kumaş Kusur Sınıflandırılmasında XGBoost Algoritmasının Kullanılması

XGBoost, yani eXtreme Gradient Boosting, yüksek performanslı bir makine öğrenimi kütüphanesidir ve özellikle sınıflandırma ve regresyon problemleri için genellikle tercih edilir. Boosting, birden fazla modelin (genellikle zayıf öğrenicilerin) bir araya getirilerek bir ansamble (topluluk) model oluşturulmasına dayanan bir makine öğrenimi tekniğidir. XGBoost, bu tekniği optimize ederek, hem hız hem de model performansı açısından mevcut en etkili ve popüler yöntemlerden biri haline gelmiştir.

XGBoost'un ana özellikleri şunlardır:

1. **Hız ve Performans**: XGBoost, yüksek performanslı bir algoritma olarak bilinir ve çok sayıda optimizasyon sayesinde eğitim süreci hızlıdır.
2. **Karburlama ve Pruning (Budama)**: XGBoost, ağaçları belli bir derinliğe kadar büyütür (karburlama) ve daha sonra kaybı azaltmayan dalları keser (budama).
3. **Düzenlileştirme**: Aşırı uyumu önlemek için L1 (Lasso) ve L2 (Ridge) düzenlileştirme tekniklerini kullanır.
4. **Esneklik**: XGBoost, sınıflandırma, regresyon, sıralama ve kullanıcı tanımlı tahmin problemleri için kullanılabilir.
5. **Ölçeklenebilirlik**: XGBoost, çok çekirdekli işlemcileri ve dağıtık bilgi işlem ortamlarını destekler, böylece büyük veri setleri üzerinde çalışabilir.
6. **Taşınabilirlik**: XGBoost, birçok platformda çalışabilir ve Hadoop, Spark ve Flink gibi büyük veri teknolojileriyle entegrasyonu destekler.
7. **Tuning (Ayarlanabilirlik)**: Modelin hiperparametrelerini ayarlayarak, çeşitli veri setlerinde en iyi performansı elde etmek için kullanıcının modeli ince ayar yapmasına olanak tanır.
8. **Kayıp Fonksiyonları**: Özelleştirilebilir kayıp fonksiyonları sayesinde, çeşitli risk modelleri veya iş gereklilikleri için özel stratejiler uygulanabilir.
9. **Çapraz Doğrulama**: Dahili olarak, modeli doğrulamak ve hiperparametre ayarlaması yapmak için çapraz doğrulama yöntemlerini destekler.
10. **Kolay Kullanım**: Python, R gibi dillerde sağlam API'ler sunar ve Pandas veri yapıları ile kolayca entegre olur.

Tablo 3. XGBoost Metrik Sonuçları

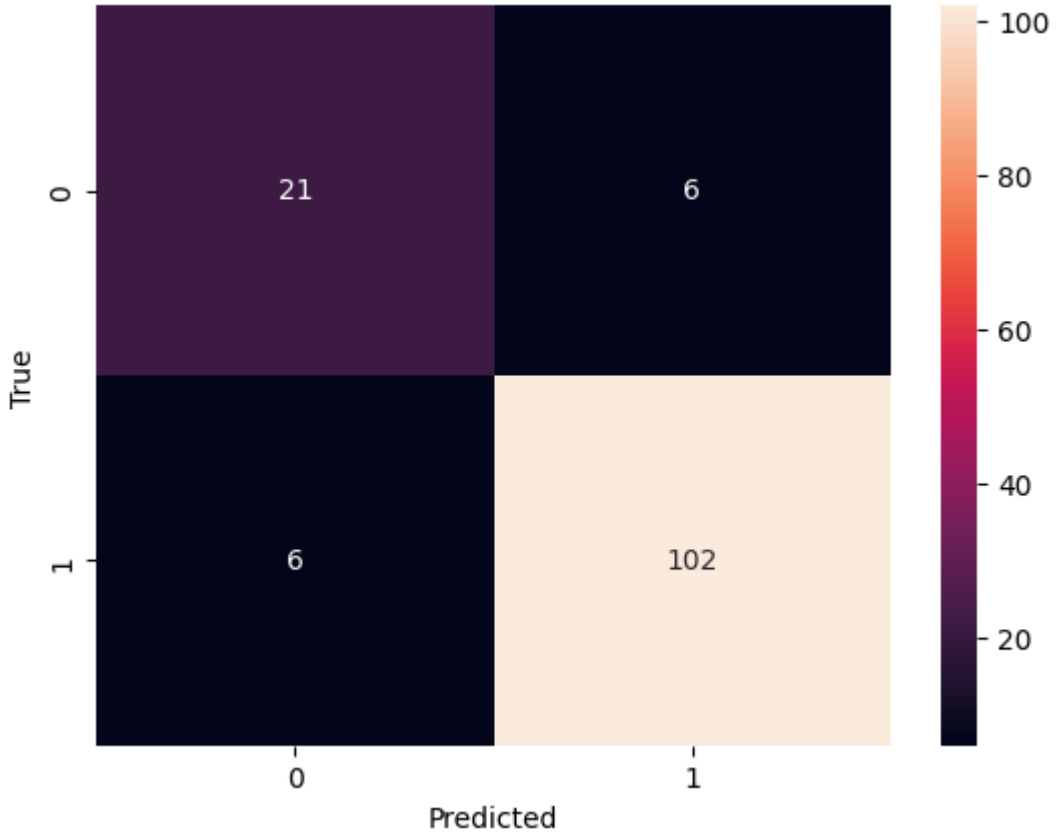
	Precision	Recall	F1-score	Support
0.0	0.78	0.78	0.78	27
1.0	0.94	0.94	0.94	108
macro avg	0.86	0.86	0.86	135
weighted avg	0.91	0.91	0.91	135
accuracy			0.91	135

Tablodaki değerler, bir XGBoost sınıflandırma modelinin performans metriklerini gösteriyor. İşte metriklerin yorumları:

1. **Class 0.0 için:**
    - **Precision (Kesinlik):** 0.78, bu, modelin sınıfı 0.0 olarak tahmin ettiği örneklerin %78'inin gerçekten sınıf 0.0 olduğunu gösterir. Yani modelin bu sınıftaki tahminlerinde %78 doğruluk oranına sahiptir.
    - **Recall (Duyarlılık):** 0.78, modelin gerçekte sınıf 0.0 olan örneklerin %78'ini doğru bir şekilde tespit ettiğini belirtir.
    - **F1-Score:** 0.78, precision ve recall'un dengeli bir ölçüsüdür ve her iki metrik arasındaki dengenin iyi olduğunu gösterir.
    - **Support:** Modelin bu sınıflandırma için 27 örneği değerlendirdiğini gösterir.
2. **Class 1.0 için:**
    - **Precision:** 0.94, bu sınıfta modelin tahminlerinin %94'ünün doğru olduğunu belirtir.
    - **Recall:** 0.94, modelin gerçekte sınıf 1.0 olan örneklerin %94'ünü doğru bir şekilde tespit ettiğini belirtir.
    - **F1-Score:** 0.94, bu sınıfta da yüksek bir F1 skoru, modelin kesinlik ve duyarlılık arasında iyi bir dengede olduğunu gösterir.
    - **Support:** 108 örnek bu sınıfta değerlendirilmiştir.
3. **Genel Metrikler:**
    - **Macro Average:** Tüm sınıfların eşit derecede önemsendiği bir ortalama hesaplamasıdır ve her bir sınıf için metriklerin basit ortalamasını alır. Burada, 0.86'lık bir makro ortalama, modelin her iki sınıf üzerinde de oldukça iyi performans gösterdiğini belirtir.
    - **Weighted Average:** Her bir sınıfın destek sayısına göre ağırlıklandırılmış ortalamasıdır. 0.91'lik ağırlıklı ortalama, modelin destek sayısını dikkate alarak genel olarak yüksek performans gösterdiğini gösterir.
4. **Genel Accuracy (Doğruluk):** 0.91, bu modelin toplamda %91 oranında doğru tahminler yaptığını belirtir ve genel bir performans göstergesidir.

Bu metrikler, XGBoost modelimizin iyi bir sınıflandırma performansı gösterdiğini belirtir. Özellikle, sınıf 1.0'da yüksek precision ve recall değerleri, modelin pozitif örnekleri tespit etmede oldukça başarılı olduğunu gösterir. Bununla birlikte, sınıf 0.0 için metrikler biraz daha düşük, bu da modelin negatif örnekleri sınıflandırmada biraz daha zorlandığını gösteriyor olabilir. Yine de, genel olarak model oldukça dengeli bir performans sergiliyor.

Grafik 4. XGBoost Confusion Matrix



Yukarıdaki görsel, bir karışıklık matrisini (confusion matrix) temsil ediyor. Karışıklık matrisi, bir sınıflandırma modelinin performansını değerlendirmek için kullanılan bir araçtır ve modelin her bir sınıf için ne kadar iyi tahminde bulunduğunu gösterir. Matriste, gerçek sınıflar satırlarda ve modelin tahminleri sütunlarda gösterilir.

Karışıklık matrisindeki değerler şu şekildedir:

- **True Positive (TP)**: Modelin pozitif olan örnekleri doğru bir şekilde pozitif olarak sınıflandırdığı örnek sayısı. Bu örnekte, 102 örnek doğru pozitif olarak belirlenmiş.
- **True Negative (TN)**: Modelin negatif olan örnekleri doğru bir şekilde negatif olarak sınıflandırdığı örnek sayısı. Bu örnekte, 21 örnek doğru negatif olarak belirlenmiş.
- **False Positive (FP)**: Modelin negatif olan örnekleri yanlışlıkla pozitif olarak sınıflandırdığı örnek sayısı. Bu örnekte, 6 örnek yanlış pozitif olarak belirlenmiş.
- **False Negative (FN)**: Modelin pozitif olan örnekleri yanlışlıkla negatif olarak sınıflandırdığı örnek sayısı. Bu örnekte, 6 örnek yanlış negatif olarak belirlenmiş.

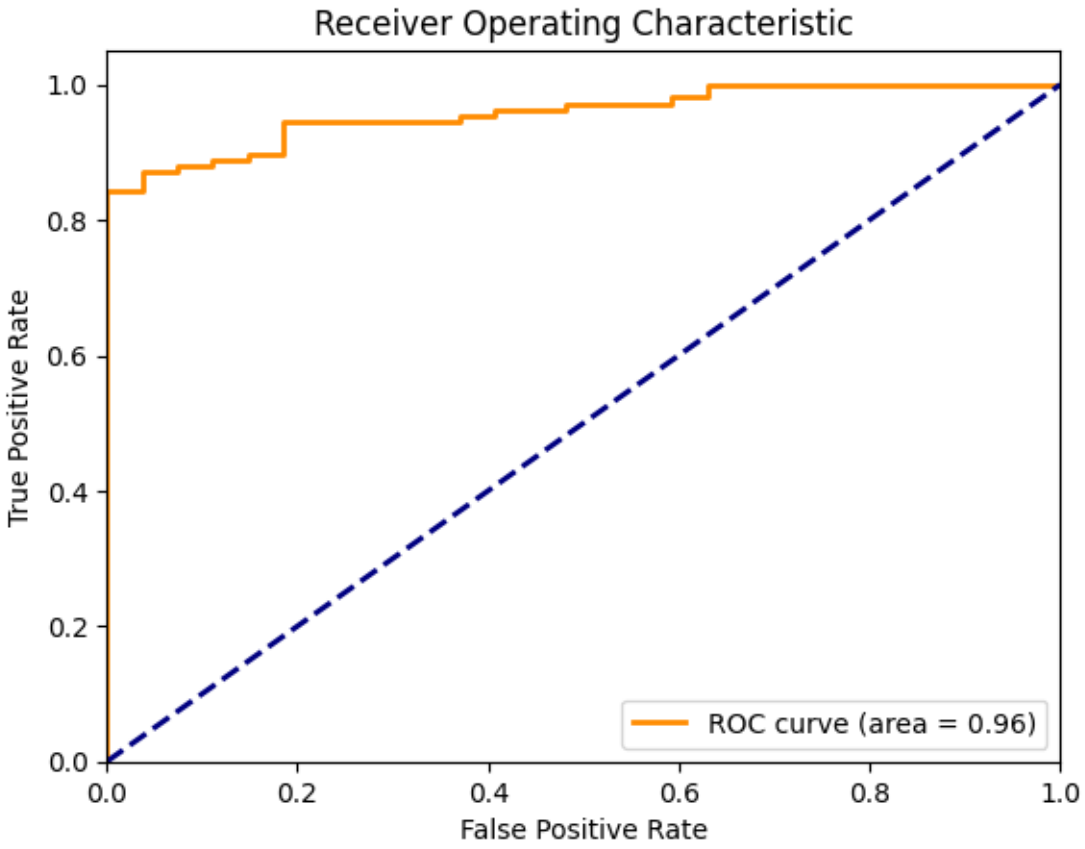
Bu değerlere dayanarak, modelin genel performansı şu şekilde yorumlanabilir:

- Modelin duyarlılığı (recall) pozitif sınıf için oldukça yüksek, çünkü 108 pozitif örnekten 102'sini doğru sınıflandırmış. Duyarlılık, TP / (TP + FN) formülüyle hesaplanır ve bu durumda yaklaşık 0.94 (102 / (102 + 6)).
- Modelin kesinliği (precision) de pozitif sınıf için yüksek, çünkü 108 tahmin edilen pozitif örneğin 102'si gerçekten pozitif. Kesinlik, TP / (TP + FP) formülüyle hesaplanır ve bu durumda 0.94 (102 / (102 + 6)).
- Negatif sınıf için, modelin hem duyarlılığı hem de kesinliği daha düşük. Bu, modelin negatif sınıfı sınıflandırmada biraz daha zorlandığını gösteriyor olabilir.

Modelin genel doğruluğu (accuracy), tüm doğru tahminlerin (TP + TN) toplam örnek sayısına (TP + FP + TN + FN) bölünmesiyle hesaplanır ve bu durumda yaklaşık 0.91 ((102 + 21) / 135) olur.

Genel olarak, model pozitif sınıfı tespit etmede iyi performans göstermiş, ancak negatif ve pozitif sınıflar arasında dengeli bir performans sergilemektedir.

Grafik 5. XGBoost ROC Grafiği



ROC eğrisi şunları gösteriyor:

- AUC değeri 0.96'dır, bu da modelimizin mükemmel bir ayrımcılık kapasitesine sahip olduğunu gösterir. Yüksek AUC değeri, modelin pozitif ve negatif sınıfları ayırt etme konusunda yüksek bir doğrulukla performans gösterdiğini gösterir.
- ROC eğrisi, mavi çizgili çapraz çizgiye (rastgele sınıflandırıcı çizgisi) göre önemli ölçüde yukarıdadır. Bu, modelimizin pozitif sınıfı tespit etmede rastgele bir tahminden çok daha iyi olduğunu gösterir.
- Eğri, sol üst köşeye doğru eğimli olduğundan, modelin düşük yanlış pozitif oranlarında bile yüksek doğru pozitif oranları elde ettiğini gösterir, bu da dengeli bir duyarlılık ve özgüllük dengesine işaret eder.

Sonuç olarak, ROC eğrisi ve AUC değeri, modelimizin test veri setindeki örnekleri sınıflandırmada yüksek bir güvenilirlik düzeyine sahip olduğunu göstermektedir.

Grafik 5. XGBoost Log Loss Grafiği

Grafikteki log kaybının analizi şu şekildedir:

- **Hızlı Düşüş:** Grafikte görüldüğü üzere, log kaybı ilk epochlarda hızlıca düşmüş. Bu, modelin eğitim veri setindeki örüntüleri hızla öğrendiğini ve büyük hatalardan çabucak kurtulduğunu gösterir.
- **İstikrar:** Daha sonra, log kaybı değeri daha yavaş bir düşüş göstermiş ve belirli bir seviyede istikrar kazanmış. Bu istikrar, modelin veri setindeki öğrenilebilir örüntüleri yakaladığını ve ekstra eğitimle daha az hata yaptığını gösterir.
- **Düşük Değer:** Grafikteki log kaybının düşük değerleri, modelin sınıflandırma görevinde iyi performans gösterdiğine işaret eder. Düşük log kaybı değerleri, modelin doğru sınıflandırma yapma olasılığının yüksek olduğunu belirtir.

Genel olarak, bu grafik modelin eğitim sürecinin sağlıklı olduğunu ve modelin, eğitim süresince veri setindeki örüntüleri başarıyla öğrendiğini gösterir.

3.5 Kumaş Kusur Sınıflandırılmasında AdaBoost Algoritmasının Kullanılması

AdaBoost (Adaptive Boosting) bir ansamble (topuluk) öğrenme algoritmasıdır ve bir dizi zayıf öğreniciyi (genellikle basit karar ağaçları) kullanarak güçlü bir öğrenici oluşturmayı amaçlar. AdaBoost, sınıflandırma ve regresyon problemleri için kullanılabilir, ancak özellikle ikili sınıflandırma problemlerinde popülerdir.

AdaBoost'un temel işleyişi, veri setindeki her örneğe eşit ağırlık vererek başlar ve bir dizi iterasyon (veya "round") boyunca aşağıdaki adımları takip eder:

1. **Zayıf Öğrenici Eğitimi**: İlk zayıf öğrenici, veri setini sınıflandırmak için eğitilir. Zayıf öğrenici, genellikle sadece bir karar düğümü (stump) olan bir karar ağacıdır.
2. **Hata Oranının Hesaplanması**: Zayıf öğrenicinin hata oranı (yanlış sınıflandırılan örneklerin ağırlıklı toplamı) hesaplanır.

3. **Öğrenici Ağırlığının Güncellenmesi**: Her öğreniciye, hata oranına bağlı olarak bir ağırlık atanır. Daha düşük hata oranına sahip öğreniciler daha yüksek ağırlık alır.
4. **Örnek Ağırlıklarının Güncellenmesi**: Yanlış sınıflandırılan örneklerin ağırlıkları artırılır, böylece sonraki öğrenici bu örnekleri doğru sınıflandırmayı öğrenmeye daha fazla odaklanır.
5. **Yeni Öğrenici Ekleme**: Yeni bir zayıf öğrenici, güncellenen ağırlıklarla eğitilir ve süreç, belirlenen iterasyon sayısı kadar veya modelin performansı kabul edilebilir bir seviyeye ulaşana kadar devam eder.
6. **Son Modelin Oluşturulması**: Tüm zayıf öğrenicilerin ağırlıklı oyları ile bir ansamble model oluşturulur.

AdaBoost'un avantajları şunlardır:

- **Otomatik Özellik Seçimi**: Zayıf öğreniciler sırasında, en iyi özellikler otomatik olarak seçilir.
- **Aşırı Uyuma Direnci**: Doğru parametrelerle, AdaBoost aşırı uyuma karşı oldukça dirençlidir.
- **Az Parametre**: Algoritma, öğrenme hızı gibi sadece birkaç parametre gerektirir ve bu parametreler genellikle geniş bir aralıkta iyi performans gösterir.
- **Yorumlanabilirlik**: Karar ağaçları kullanıldığında, modelin nihai kararları yorumlamak nispeten kolaydır.

Ancak AdaBoost'un bazı dezavantajları da vardır:

- **Gürültülü Veri ve Aykırı Değerlere Duyarlılık**: Yanlış sınıflandırılan örneklerin ağırlıkları arttığı için, gürültülü veri ve aykırı değerler modeli yanıltabilir.
- **Zaman ve Hesaplama Maliyeti**: Büyük veri setlerinde ve çok sayıda iterasyonda zaman alıcı olabilir.

AdaBoost, özellikle karar sınırlarının karmaşık olmadığı ve veri setinin gürültüden nispeten arınmış olduğu durumlarda oldukça etkilidir.

Tablo 3. AdaBoost Metrik Sonuçları

	Precision	**Recall**	**F1-score**	**Support**
0.0	0.71	0.74	0.73	27
1.0	0.93	0.93	0.93	108
macro avg	0.82	0.83	0.83	135
weighted avg	0.89	0.89	0.89	135
accuracy			0.89	135

Yukarıdaki AdaBoost modelinin performans metrikleri tablosu şu şekilde yorumlanabilir:

1. **Class 0.0 için:**
    - **Precision (Kesinlik):** 0.71, bu modelin sınıf 0.0 olarak tahmin ettiği örneklerin %71'inin gerçekten sınıf 0.0 olduğunu gösterir. Bu, modelin bu sınıftaki tahminlerinin nispeten doğru olduğunu, ancak bazı yanlış pozitiflerin (yanlışlıkla negatif olarak işaretlenen pozitif örnekler) olduğunu gösterir.
    - **Recall (Duyarlılık):** 0.74, modelin gerçekte sınıf 0.0 olan örneklerin %74'ünü doğru bir şekilde tespit ettiğini belirtir. Modelin bu sınıftaki örnekleri kaçırmadan tanıma kabiliyeti iyi görünmektedir.

- **F1-Score:** 0.73, kesinlik ve duyarlılığın dengeli bir ölçüsüdür ve her iki metrik arasında dengeli bir performans olduğunu gösterir.

2. **Class 1.0 için:**

    - **Precision:** 0.93, bu modelin sınıf 1.0 olarak tahmin ettiği örneklerin %93'ünün gerçekten sınıf 1.0 olduğunu gösterir. Bu, modelin bu sınıftaki tahminlerinde oldukça kesin olduğunu gösterir.
    - **Recall:** 0.93, modelin gerçekte sınıf 1.0 olan örneklerin %93'ünü doğru bir şekilde tespit ettiğini belirtir. Modelin bu sınıftaki örnekleri kaçırmadan tanıma kabiliyeti yüksek görünmektedir.
    - **F1-Score:** 0.93, bu sınıfta da yüksek bir F1 skoru, modelin kesinlik ve duyarlılık arasında mükemmel bir dengede olduğunu gösterir.

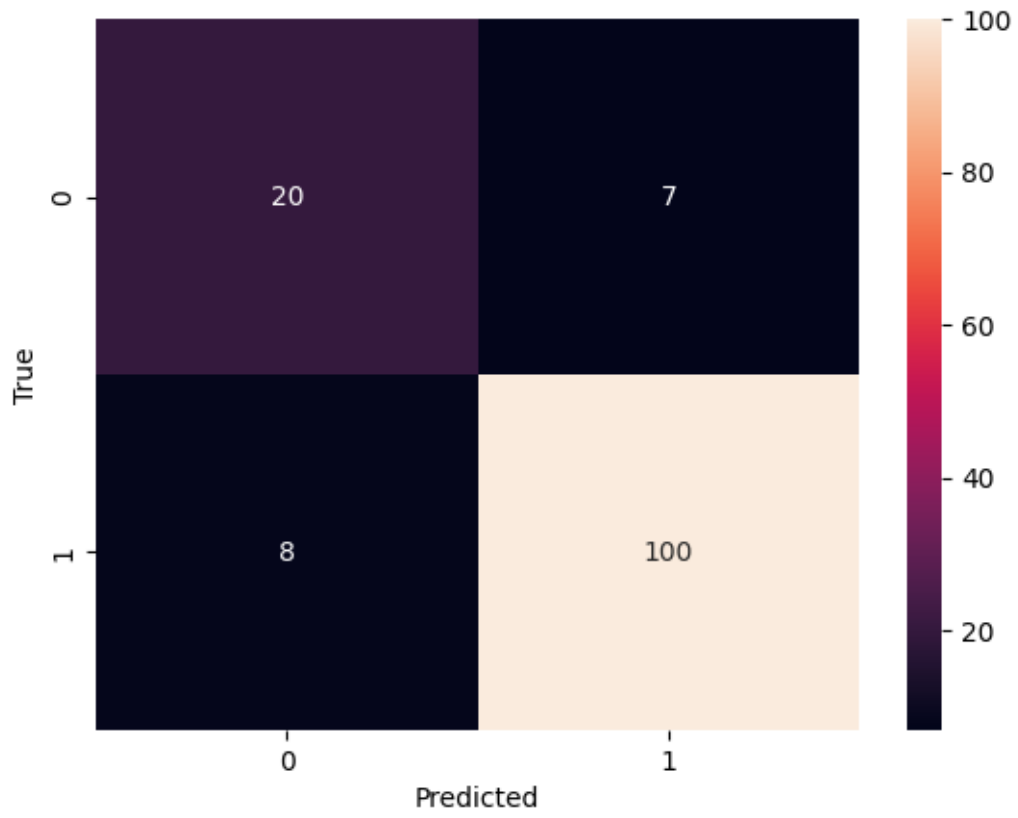
3. **Genel Metrikler:**

    - **Macro Average:** Her sınıf için metriklerin basit ortalamasını alır ve 0.82 (kesinlik), 0.83 (duyarlılık) ve 0.83 (F1 skoru) değerlerini verir. Bu, modelin her iki sınıf üzerinde de dengeli bir performans gösterdiğini gösterir.
    - **Weighted Average:** Modelin her sınıfın destek sayısını dikkate alarak hesaplanmış ağırlıklı ortalamalarıdır. Bu durumda, 0.89 (kesinlik), 0.89 (duyarlılık) ve 0.89 (F1 skoru) değerleri modelin genel olarak yüksek ve dengeli bir performans gösterdiğini gösterir.

4. **Genel Accuracy (Doğruluk):** 0.89, bu oran modelin tüm tahminlerin %89'unu doğru yaptığını belirtir ve genel performansın iyi olduğunu gösterir.

Bu metrikler, AdaBoost modelimizin her iki sınıfı da oldukça iyi bir şekilde ayırt edebildiğini gösteriyor. Özellikle, sınıf 1.0 için yüksek precision ve recall değerleri, modelin pozitif örnekleri çok iyi tespit edebildiğini gösteriyor. Bununla birlikte, sınıf 0.0 için precision ve recall biraz daha düşük, bu da modelin negatif örnekleri tespit etmede biraz daha fazla zorluk yaşadığını gösterebilir. Her iki sınıf için de F1-skorları oldukça yüksek, bu da modelin her iki sınıfı da dengeli bir şekilde tespit edebildiğini gösterir.

Grafik 6. AdaBoost Confusion Matrix



Yukarıdaki görsel, AdaBoost modelimizin karışıklık matrisini (confusion matrix) gösteriyor. Karışıklık matrisi, modelimizin sınıflandırma performansını ayrıntılı bir şekilde gösterir ve gerçek sınıflar ile modelin tahminleri arasındaki ilişkiyi sayısal olarak ifade eder.

Bu karışıklık matrisi şu değerlere sahiptir:

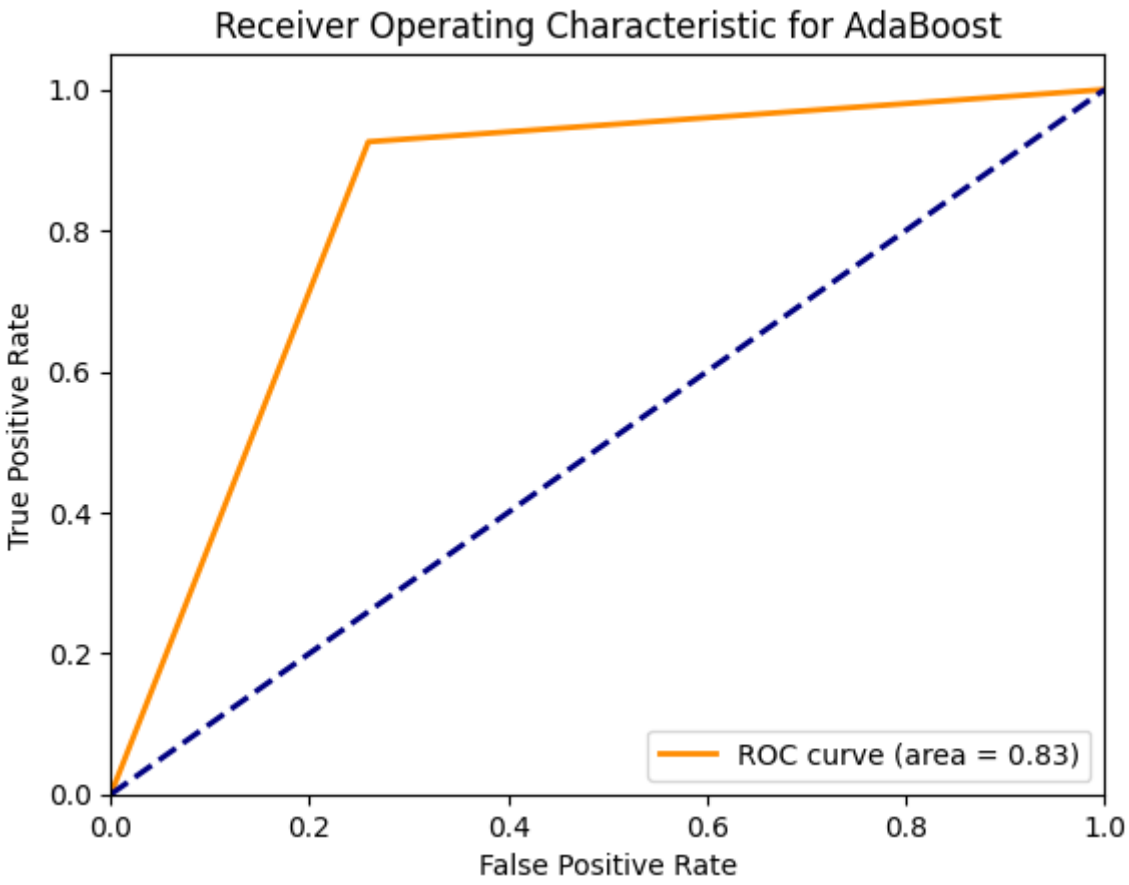
- **True Negative (TN)**: 20, modelin gerçekte negatif (0) olan ve negatif olarak tahmin edilen örnek sayısı.
- **False Positive (FP)**: 7, modelin gerçekte negatif (0) olan ancak pozitif (1) olarak yanlış tahmin edilen örnek sayısı.
- **False Negative (FN)**: 8, modelin gerçekte pozitif (1) olan ancak negatif (0) olarak yanlış tahmin edilen örnek sayısı.
- **True Positive (TP)**: 100, modelin gerçekte pozitif (1) olan ve pozitif olarak doğru tahmin edilen örnek sayısı.

Bu değerlerden yola çıkarak şu yorumları yapabiliriz:

- Modelin duyarlılığı veya geri çağırma oranı (recall) için pozitif sınıf oldukça yüksek: TP / (TP + FN) = 100 / (100 + 8) = yaklaşık 0.93. Bu, modelin pozitif sınıfı tespit etme konusunda oldukça başarılı olduğunu gösterir.
- Pozitif sınıf için kesinlik (precision) da oldukça yüksek: TP / (TP + FP) = 100 / (100 + 7) = yaklaşık 0.93. Bu, modelin pozitif olarak işaretlediği örneklerin büyük çoğunluğunun gerçekten pozitif olduğunu gösterir.
- Negatif sınıf için, modelin performansı biraz daha düşük. TN / (TN + FP) = 20 / (20 + 7) = yaklaşık 0.74 ile negatif örnekleri doğru bir şekilde tespit etme oranı daha düşüktür.
- Genel doğruluk (accuracy), tüm doğru tahminlerin toplam örnek sayısına oranıdır ve bu durumda (TP + TN) / (TP + FP + TN + FN) = (100 + 20) / 135 = yaklaşık 0.89 olur.

Karışıklık matrisi, modelin genel olarak iyi performans gösterdiğini ancak negatif sınıfı tespit etme konusunda biraz daha zorlandığını göstermektedir. Özellikle, modelin negatif sınıf üzerindeki duyarlılığını (recall) ve kesinliğini (precision) artırmaya yönelik iyileştirmeler yapılabilir. Bununla birlikte, genel olarak modelin pozitif sınıfı tespit etmede başarılı olduğu ve toplamda yüksek doğruluk sağladığı görülmektedir.

Grafik 7. AdaBoost ROC Grafiği



Yukarıdaki görsel, AdaBoost modelimiz için bir ROC (Receiver Operating Characteristic) eğrisini göstermektedir. ROC eğrisi, farklı eşik seviyelerinde modelimizin doğru pozitif oranı (TPR) ile yanlış pozitif oranı (FPR) arasındaki ilişkiyi gösterir. Eğrinin altında kalan alanın (AUC - Area Under Curve) büyüklüğü, modelin sınıflandırma performansını niceliksel olarak gösterir.

Yüklenen ROC eğrisine göre şu yorumları yapabiliriz:

- **AUC Değeri:** AUC 0.83 olarak belirtilmiş. Bu değer, modelimizin rastgele bir sınıflandırıcıdan önemli ölçüde daha iyi performans gösterdiğini ve pozitif ile negatif sınıfları ayırt etme yeteneğinin oldukça iyi olduğunu gösterir. AUC değeri 1'e yakın olduğunda, modelin mükemmel sınıflandırma performansına sahip olduğunu ifade eder. 0.83, iyi bir değerdir ancak mükemmel değerlerden (genellikle 0.9 ve üzeri) biraz daha düşüktür.

- **Eğri Şekli:** ROC eğrisi genel olarak mavi çizgili çapraz çizginin (rastgele tahmin çizgisi) üzerindedir, bu da modelin pozitif ve negatif sınıfları ayırt etme konusunda belirgin bir kapasiteye sahip olduğunu gösterir. Ancak, eğrinin bazı kısımlarında düz bir çizgiye dönüşerek TPR'de bir artış olmadan FPR'nin arttığı görülüyor. Bu, belirli eşik değerlerinde modelin yanlış pozitifleri artırdığı ancak buna karşılık doğru pozitifleri artıramadığı anlamına gelebilir.

Genel olarak, modelimizin AdaBoost kullanarak iyi bir sınıflandırma kapasitesine sahip olduğunu, ancak mükemmellikten bir miktar uzak olduğunu göstermektedir.

3.6 Kumaş Kusur Sınıflandırılmasında Support Vector Machine (SVM) Algoritmasının Kullanılması

Support Vector Machine (SVM), gözetimli öğrenme kapsamında kullanılan popüler ve güçlü bir makine öğrenimi algoritmasıdır. Hem sınıflandırma hem de regresyon görevlerinde kullanılabilen SVM, özellikle sınıflandırma problemlerinde yaygın olarak tercih edilir. SVM, veri setindeki örnekleri ayırt etmek için bir veya daha fazla hiper-düzlem (karar sınırı) kullanır. Temel amaç, farklı sınıflar arasında en iyi ayrımı sağlayan hiper-düzlemi bulmaktır.

SVM'nin Temel Kavramları:

- **Hiper-Düzlem**: SVM, farklı sınıfları ayıran optimal hiper-düzlemi bulmaya çalışır. Hiper-düzlem, n-boyutlu uzayda (n, örneklerin özellik sayısı) bir düzlemdir.
- **Marjin**: İki sınıf arasındaki en geniş şerit veya marjini bulmayı amaçlar. SVM, bu marjini maksimize eden hiper-düzlemi seçer.
- **Destek Vektörleri**: Her iki sınıfa ait örneklerden en yakın olanları ve hiper-düzlemin konumunu belirleyen örneklerdir. Bu örnekler, hiper-düzlemin konumunu belirleyen anahtar örneklerdir.
- **Kernel Trick**: Doğrusal olarak ayrılamayan veri setlerinde, örnekleri daha yüksek boyutlu bir uzaya dönüştürmek için kullanılır. Böylece, doğrusal olmayan problemler de SVM ile çözülebilir hale gelir.

SVM'nin Avantajları:

- Güçlü ve esnektir, çeşitli karmaşık problemleri çözmede etkilidir.
- Overfitting (aşırı öğrenme) problemlerine karşı dirençlidir.
- Az sayıda destek vektörü kullanarak karar fonksiyonunu ifade eder, bu da modelin hem hızlı hem de bellek açısından verimli olmasını sağlar.

SVM'nin Dezavantajları:

- Doğru bir SVM modeli oluşturmak için özellik seçimi ve kernel seçimi gibi hiperparametre ayarlamaları kritiktir.
- Büyük veri setlerinde eğitim süresi uzun olabilir.
- Çıktıları yorumlamak diğer bazı algoritmalara göre daha zordur çünkü karar sınırları karmaşık olabilir ve doğrudan anlaşılır bir formülle ifade edilmez.

SVM Uygulamaları:

- Yüz tanıma, el yazısı tanıma ve görüntü sınıflandırma gibi görüntü işleme uygulamaları.
- Metin sınıflandırma, spam filtreleme ve duygu analizi gibi metin ve doğal dil işleme görevleri.
- Biyoenformatikte, kanser tespiti veya protein sınıflandırma gibi biyolojik ve kimyasal problemler.

SVM Modellerini Eğitirken Dikkat Edilmesi Gerekenler:

- Modelinizi eğitmeden önce veri ön işleme çok önemlidir. Özelliklerin ölçeklendirilmesi (normalizasyon veya standartlaştırma) genellikle performansı iyileştirir.
- Çekirdek fonksiyonun seçimi (lineer, polinom, RBF vb.), modelin performansını büyük ölçüde etkiler.
- SVM, sınıflandırma görevlerinde olasılık tahminleri sağlamak için ek bir lojistik regresyon katmanı gerektirebilir, çünkü temel SVM sınıflandırıcısı sadece sınıfları ayırır, olasılık değerleri vermez.

SVM, modern makine öğrenimi uygulamalarında halen önemli bir yere sahiptir ve birçok alanda başarılı sonuçlar üretebilen güçlü bir araç olarak kabul edilir.

Tablo 4. Support Vector Machine Metrik Sonuçları

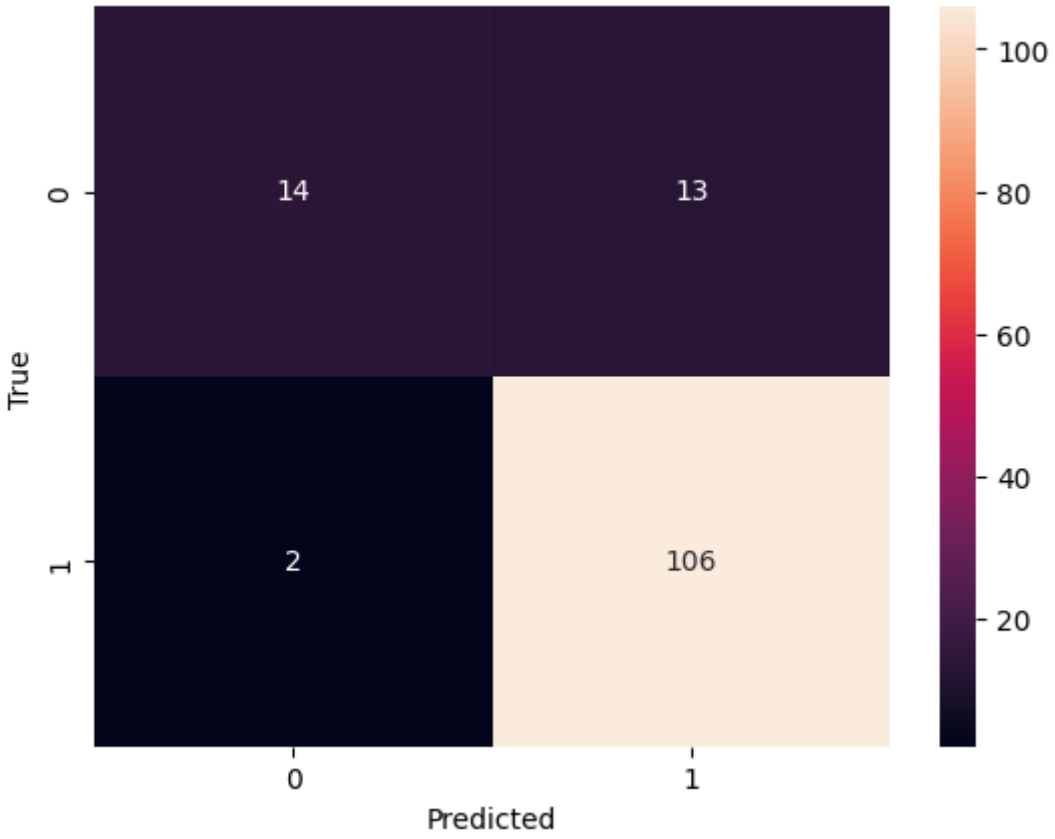
	Precision	Recall	F1-score	Support
0.0	0.88	0.52	0.65	27
1.0	0.89	0.98	0.93	108
macro avg	0.88	0.75	0.79	135
weighted avg	0.89	0.89	0.88	135
accuracy			0.89	135

Yukarıdaki tablo Destek Vektör Makinesi (SVM) sınıflandırma modelinin performansını gösteriyor. İşte bu metriklerin yorumları:

1. **Class 0.0 için:**
    - **Precision (Kesinlik):** 0.88, modelin sınıf 0.0 olarak tahmin ettiği örneklerin %88'inin gerçekten sınıf 0.0 olduğunu gösterir.
    - **Recall (Duyarlılık):** 0.52, modelin gerçekte sınıf 0.0 olan örneklerin %52'sini doğru bir şekilde tespit ettiğini belirtir.
    - **F1-Score:** 0.65, precision ve recall'un dengeli bir ölçüsüdür ve bu sınıf için düşük bir değer olduğunu gösterir, bu da modelin bu sınıftaki örnekleri hem doğru tahmin etmede hem de kaçırmada zorlandığını gösterir.
2. **Class 1.0 için:**
    - **Precision:** 0.89, modelin sınıf 1.0 olarak tahmin ettiği örneklerin %89'unun gerçekten sınıf 1.0 olduğunu gösterir.
    - **Recall:** 0.98, modelin gerçekte sınıf 1.0 olan örneklerin %98'ini doğru bir şekilde tespit ettiğini belirtir.
    - **F1-Score:** 0.93, bu sınıfta yüksek bir F1 skoru, modelin kesinlik ve duyarlılık arasında mükemmel bir dengede olduğunu gösterir.
3. **Genel Metrikler:**
    - **Macro Average:** Her sınıf için metriklerin basit ortalamasını alır ve 0.88 (kesinlik), 0.75 (duyarlılık) ve 0.79 (F1 skoru) değerlerini verir. Bu, modelin her iki sınıf üzerinde dengeli performans göstermediğini gösterir, çünkü sınıf 0.0 için düşük recall ve F1 skoru var.
    - **Weighted Average:** Modelin her sınıfın destek sayısını dikkate alarak hesaplanmış ağırlıklı ortalamalarıdır. Bu durumda, 0.89 (kesinlik), 0.89 (duyarlılık) ve 0.88 (F1 skoru) değerleri modelin genel olarak yüksek ve destek sayısına göre dengeli bir performans gösterdiğini gösterir.
4. **Genel Accuracy (Doğruluk):** 0.89, bu oran modelin tüm tahminlerin %89'unu doğru yaptığını belirtir ve genel performansın iyi olduğunu gösterir.

Bu metrikler, SVM modelimizin özellikle pozitif sınıfı (1.0) tespit etmede çok iyi performans gösterdiğini, ancak negatif sınıfı (0.0) tespit etmede bazı zorluklar yaşadığını gösteriyor.

Grafik 8. Support Vector Machine Confusion Matrix



Yukarıdaki görsel, Destek Vektör Makinesi (SVM) modelimizin karışıklık matrisini göstermektedir. Karışıklık matrisi, modelin gerçek sınıflar ve tahmin edilen sınıflar arasındaki performansını gösteren dört kısımdan oluşur.

Karışıklık matrisinde yer alan değerler:

- **True Negative (TN):** 14. Modelin negatif olarak doğru bir şekilde sınıflandırdığı gerçek negatif örnek sayısı.
- **False Negative (FN):** 2. Modelin pozitif olarak yanlış bir şekilde sınıflandırdığı gerçek negatif örnek sayısı.
- **True Positive (TP):** 106. Modelin pozitif olarak doğru bir şekilde sınıflandırdığı gerçek pozitif örnek sayısı.
- **False Positive (FP):** 13. Modelin negatif olarak yanlış bir şekilde sınıflandırdığı gerçek pozitif örnek sayısı.

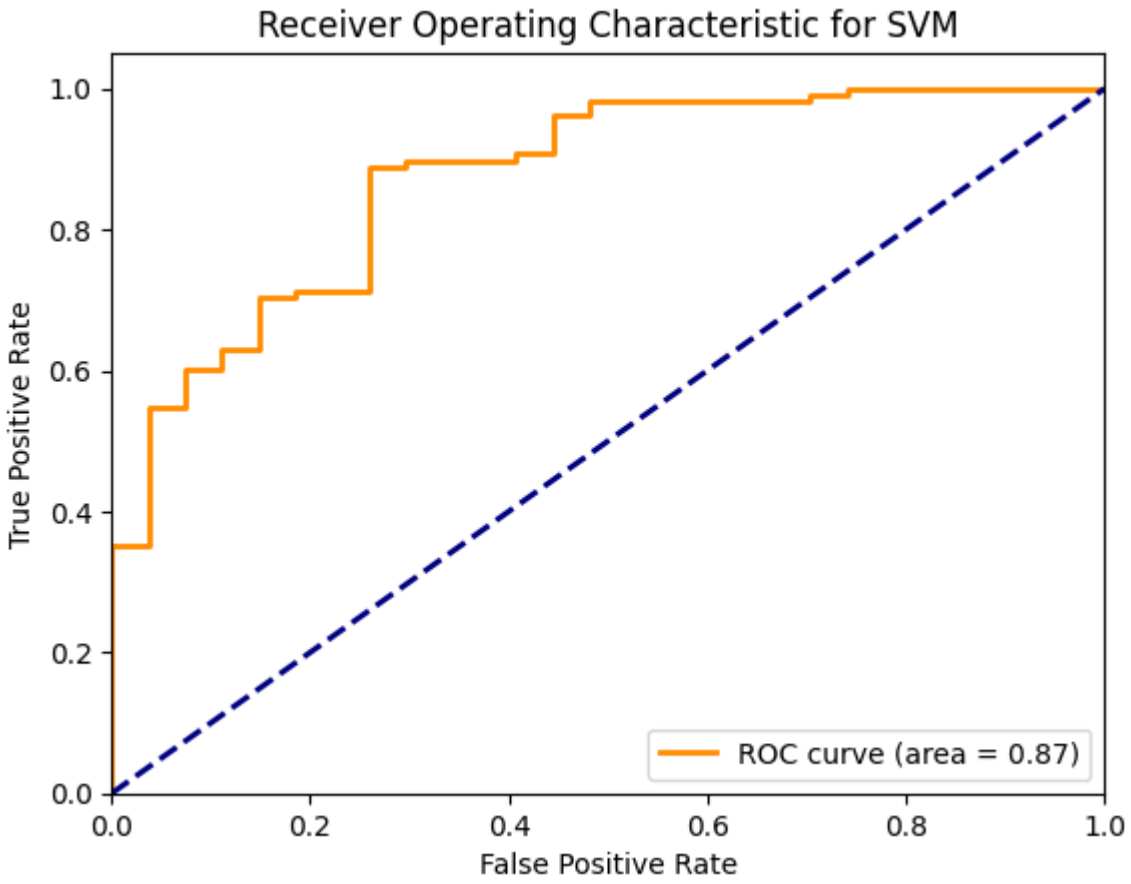
Bu değerlerden yola çıkarak modelin performansını şu şekilde yorumlayabiliriz:

- Model, pozitif sınıfı (1) tespit etmede oldukça başarılı görünüyor çünkü gerçek pozitiflerin çoğunu doğru bir şekilde sınıflandırabilmiş (106 doğru pozitif). Ancak, negatif sınıfı (0) tespit etme konusunda zorlanıyor olabilir, çünkü negatif sınıfın neredeyse yarısını yanlış pozitif olarak sınıflandırmış (13 yanlış pozitif).
- Modelin duyarlılığı (recall) veya geri çağırma oranı pozitif sınıf için yüksek (TP / (TP + FN) = 106 / (106 + 2) ≈ 0.98), bu da modelin pozitif örnekleri kaçırmadan tespit edebildiğini gösteriyor.

- Pozitif sınıf için modelin kesinliği (precision) (TP / (TP + FP) = 106 / (106 + 13) ≈ 0.89) da oldukça yüksektir ve modelin pozitif olarak işaretlediği örneklerin büyük çoğunluğunun gerçekten pozitif olduğunu gösterir.
- Negatif sınıf için duyarlılık düşüktür (TN / (TN + FP) = 14 / (14 + 13) ≈ 0.52), bu da modelin gerçek negatif örnekleri tespit etme konusunda zorlandığını gösterir.
- Genel doğruluk (accuracy), (TP + TN) / (TP + FP + TN + FN) = (106 + 14) / (106 + 14 + 2 + 13) ≈ 0.89, bu oran modelin genel olarak yüksek bir doğrulukla tahmin yaptığını gösterir.

Modelin genel performansı yüksek olmasına rağmen, negatif örnekleri tespit etme konusunda sıkıntılar yaşamaktadır.

Grafik 9. Support Vector Machine ROC Grafiği



Yukarıdaki görselde görünen ROC (Receiver Operating Characteristic) eğrisi, Destek Vektör Makinesi (SVM) modelimizin sınıflandırma performansını değerlendiren bir araçtır. ROC eğrisi, farklı eşik değerlerinde modelin doğru pozitif oranı (True Positive Rate, TPR) ile yanlış pozitif oranı (False Positive Rate, FPR) arasındaki ilişkiyi gösterir.

Bu ROC eğrisinin yorumu şu şekildedir:

- **AUC (Area Under the Curve) Değeri:** Eğrinin altında kalan alanın büyüklüğü 0.87'dir. Bu değer, modelimizin rastgele tahminlerden önemli ölçüde daha iyi performans gösterdiğini ve pozitif ile negatif sınıfları ayırt etme yeteneğinin oldukça iyi olduğunu gösterir. AUC değeri 1'e yakın olursa, modelin mükemmel bir sınıflandırma performansına sahip olduğunu ifade eder. 0.87, iyi bir değerdir ve genellikle iyi bir sınıflandırıcıyı işaret eder.

- **Eğri Şekli:** ROC eğrisi, sol üst köşeye doğru ilerler, bu da modelin düşük yanlış pozitif oranlarında yüksek doğru pozitif oranları elde ettiğini gösterir. Eğrinin sol üst köşeye ne kadar yakın olması, modelin o kadar iyi performans gösterdiğini belirtir.
- **Eğrinin Yükselişi:** Eğrinin bazı noktalarda yatay olarak ilerlediği, bazı noktalarda ise dikey bir yükseliş gösterdiği gözlemlenmektedir. Bu, belirli eşik değerlerinde modelin duyarlılığının arttığını ancak aynı zamanda yanlış pozitif oranının sabit kaldığını veya tam tersini gösterir.

Genel olarak, 0.87'lik AUC değeri ile modelimiz, veri setindeki sınıfları ayırt etmede etkili bir performans gösteriyor.

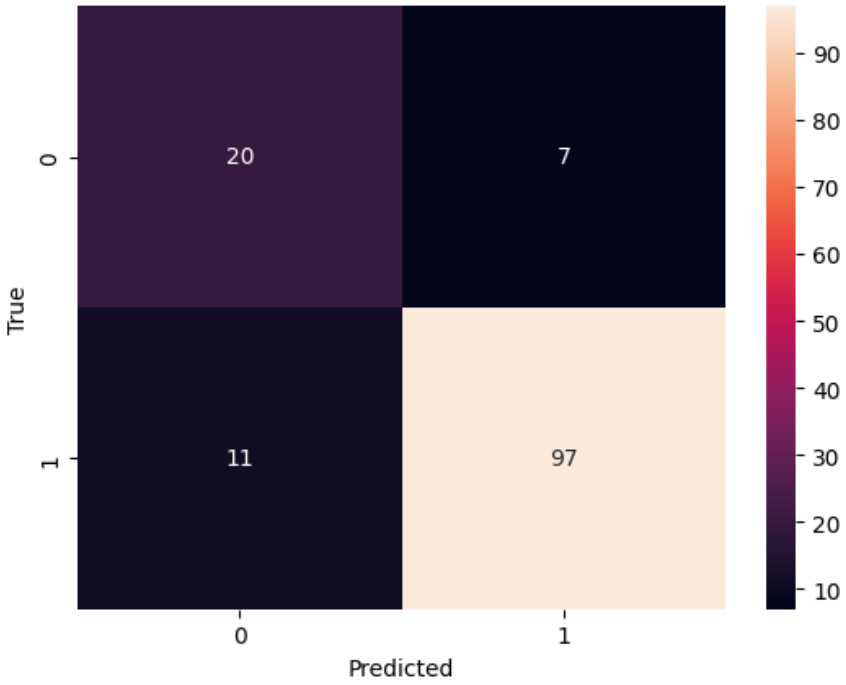
3.7 Kumaş Kusur Sınıflandırılmasında K-Nearest Neighbors (KNN) Algoritmasının Kullanılması

K-En Yakın Komşu (KNN - K-Nearest Neighbors) algoritması, gözetimli öğrenme modelleri arasında yer alır ve hem sınıflandırma hem de regresyon problemleri için kullanılabilir. Basitlik ve etkinlik açısından popüler bir algoritmadır. KNN'in temel çalışma prensibi, bir örnek verildiğinde, bu örneğe en yakın 'K' adet komşusuna bakarak bir tahmin yapmaktır. "Yakınlık" genellikle Öklid, Manhattan veya Minkowski gibi mesafe metrikleri kullanılarak hesaplanır.

KNN'in Temel Özellikleri:

- **Tembellik Tabanlı Model**: KNN, tembel bir algoritmadır çünkü eğitim aşamasında veriyi modelleme yerine hafızada saklar ve tahmin aşamasında hesaplama yapar.
- **Parametrik Olmayan Yöntem**: Veri dağılımı hakkında herhangi bir varsayımda bulunmaz ve bu nedenle parametrik olmayan bir yöntemdir.
- **Örnek Tabanlı Öğrenme**: KNN, veri noktalarını hafızada tutar ve tahmin yaparken bu örnekleri kullanır.

Grafik 10. KNN Confusion Matrix



Yukarıdaki görsel, bir K-Nearest Neighbors (KNN) sınıflandırma modelinin karışıklık matrisini temsil ediyor. Karışıklık matrisi, sınıflandırma modelinin performansını gösteren bir matristir ve modelin gerçek değerlere kıyasla tahminlerini aşağıdaki dört temel bileşen üzerinden değerlendirir:

Görseldeki karışıklık matrisindeki değerler şunlardır:

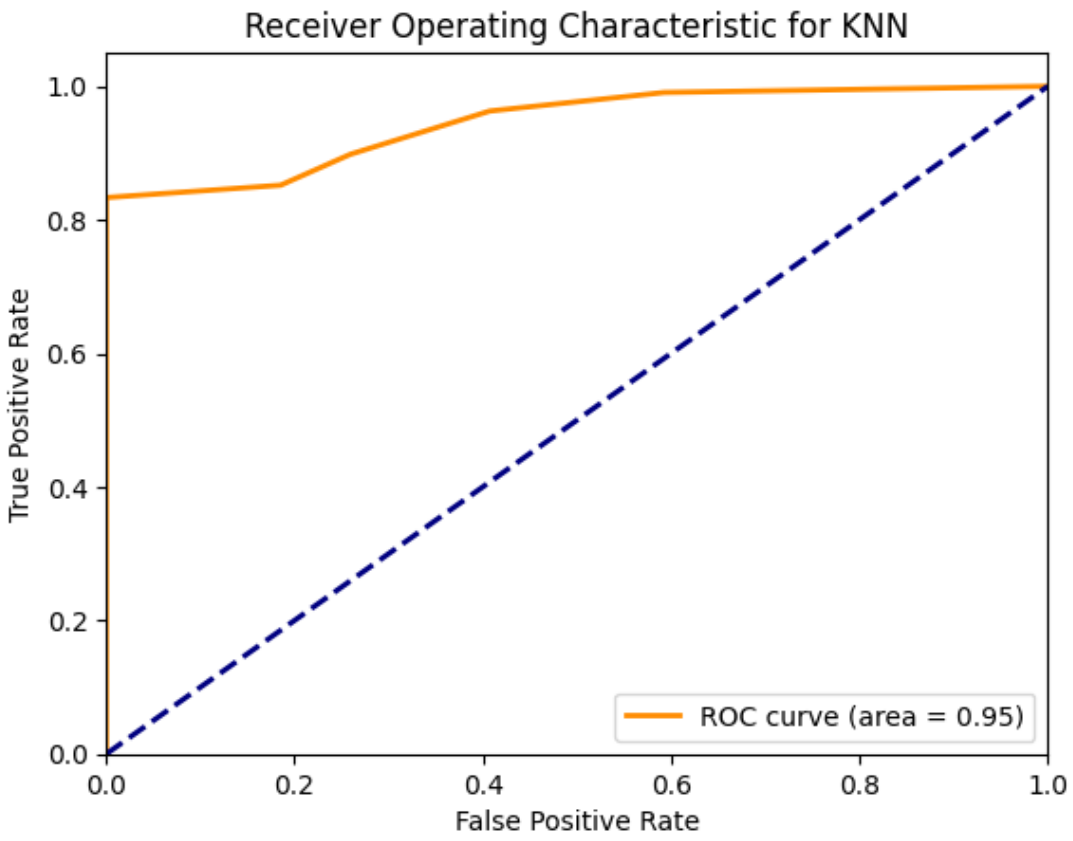
- **TN**: 20
- **FP**: 7
- **FN**: 11
- **TP**: 97

Bu değerler şu anlamı taşır:

- Model, 108 pozitif örneğin 97'sini doğru bir şekilde tespit etmiş. Bu, yüksek bir True Positive oranını gösterir ve modelin pozitif sınıfı tespit etme konusunda etkili olduğunu düşündürür.
- Negatif sınıf için model, 27 örneğin 20'sini doğru bir şekilde tespit etmiş, bu da True Negative oranını gösterir. Ancak, model 7 örneği yanlışlıkla pozitif olarak sınıflandırmış, bu da bazı False Positive hatalarını işaret eder.
- Modelin duyarlılığı (recall) veya geri çağırma oranı için pozitif sınıf, TP / (TP + FN) = 97 / (97 + 11) ≈ 0.898, yani modelin gerçek pozitiflerin yaklaşık %89.8'ini doğru bir şekilde tespit ettiği anlamına gelir.
- Kesinlik (precision) oranı, TP / (TP + FP) = 97 / (97 + 7) ≈ 0.933, yani modelin pozitif olarak sınıflandırdığı örneklerin yaklaşık %93.3'ünün gerçekten pozitif olduğunu gösterir.

Genel olarak, modelin yüksek bir doğruluk (accuracy) oranına sahip olduğu ve pozitif sınıfı iyi bir kesinlikle tespit edebildiği görülüyor. Ancak, negatif sınıfı tespit ederken bir miktar yanlış pozitif hata yapılmıştır.

Grafik 11. KNN ROC Grafiği



Yukarıdaki görselde gösterilen, K-Nearest Neighbors (KNN) algoritması için ROC (Receiver Operating Characteristic) eğrisidir. ROC eğrisi, modelin farklı eşik değerlerinde doğru pozitif oranı (True Positive Rate, TPR) ile yanlış pozitif oranı (False Positive Rate, FPR) arasındaki ilişkiyi gösterir.

ROC eğrisine göre şu yorumları yapabiliriz:

- **AUC Değeri:** AUC 0.95 olarak belirtilmiş. Bu, modelin pozitif ve negatif sınıfları ayırt etme konusunda mükemmele yakın bir performans gösterdiğini ifade eder. Yüksek AUC değeri, modelin çoğu durumda doğru tahmin yaptığını ve yanlış pozitifler ile yanlış negatifleri minimuma indirdiğini gösterir.
- **Eğri Şekli:** ROC eğrisi sol üst köşeye oldukça yakın. Bu, düşük yanlış pozitif oranlarında yüksek doğru pozitif oranları elde ettiği anlamına gelir, yani modelin yanlış alarm vermeden gerçek pozitifleri tespit etme kabiliyeti oldukça yüksektir.
- **Eğri ve Rastgele Tahmin Çizgisi Arasındaki Mesafe:** Eğri, mavi çizgili çapraz çizginin (rastgele tahmin çizgisi) oldukça üzerindedir, bu da modelin rastgele bir tahminden çok daha iyi performans gösterdiğini gösterir.

Kısacası, 0.95'lik AUC değeri ile KNN modelinizin çok iyi bir sınıflandırma performansına sahip olduğunu ve test edilen veri seti üzerinde yüksek duyarlılık ve özgüllük oranlarına ulaştığını göstermektedir.

4. Sonuç

Bu çalışmada, beş farklı makine öğrenimi algoritmasının performansı incelenmiştir: Evrişimli Sinir Ağları (CNN), eXtreme Gradient Boosting (XGBoost), Adaptive Boosting (AdaBoost), Destek Vektör Makineleri (SVM) ve K-En Yakın Komşular (KNN). Her bir modelin performansı, sınıflandırma doğruluğu, kesinlik, duyarlılık ve F1 skoru gibi metriklerle ve karışıklık matrisleri ile ROC eğrileri aracılığıyla değerlendirilmiştir. Analiz sonuçları, her bir algoritmanın güçlü ve zayıf yönlerini ortaya koymuştur.

CNN modeli, görsel veri üzerinde etkileyici bir performans göstermiş, yüksek doğruluk ve azalan kayıplarla öğrenme sürecinin istikrarını kanıtlamıştır. Bu durum, CNN'nin görüntü sınıflandırma görevleri için güçlü bir model olduğunu göstermektedir. XGBoost ve AdaBoost algoritmaları, yapılandırılmış veri üzerinde etkileyici doğruluk oranları sağlamış, ancak özellikle AdaBoost'un negatif sınıfları tespit etme konusunda sıkıntılar yaşadığı gözlemlenmiştir. SVM, marjinal sınıflandırma yeteneği ile yüksek doğruluk ve F1 skorları elde etmiş, özellikle pozitif sınıfları tespit etme konusunda kuvvetli olduğunu göstermiştir. Ancak, negatif sınıfların tespiti noktasında SVM'nin performansı, özellikle duyarlılık açısından, diğer modellere kıyasla daha düşük kalmıştır. Son olarak, KNN algoritması, tutarlı ve dengeli bir performans sergilemiş, ancak büyük veri setleri ve yüksek boyutlulukla ilgili zorluklar göz önünde bulundurulduğunda ölçeklenebilirlik sorunlarına dikkat çekmiştir.

Sonuç olarak, her algoritmanın spesifik veri setleri ve problem tanımları üzerindeki başarısı, ilgili algoritmaların doğası ve model parametrelerinin dikkatli bir şekilde ayarlanmasına bağlıdır. Pratik uygulamalarda, bu algoritmaların her birinin özellikleri ve sınırlamaları göz önünde bulundurularak özenli bir seçim yapılmalıdır. Ayrıca, algoritmaların genelleme kabiliyetlerini doğrulamak için bağımsız veri setleri üzerinde çapraz doğrulama gibi yöntemler kullanılmalıdır. Bu çalışma, makine öğrenimi algoritmalarının performanslarının kapsamlı bir değerlendirilmesine katkıda bulunmayı amaçlamaktadır ve bulgular, makine öğrenimi topluluğu için referans bir kaynak olarak hizmet edebilir.

Kodun tamamını Github linkimden bulabilirsiniz:

https://github.com/berkemaydemir/AitexMachineLearning-CNN

5. Kaynaklar

[1] Kumar, A. (2008). Computer-vision-based fabric defect detection: a survey. IEEE Transactions on Industrial Electronics, 55(1), 348-363.

[2] Ngan, H. Y., Pang, G. K., Yung, N. H. (2011). Automated fabric defect detection – a review. Image and Vision Computing, 29(7), 442-458.

[3] M. Boluki and F. Mohanna, "Inspection of textile fabrics based on the optimal gabor filter," Signal, Image and Video Processing, vol. 15, no. 7, pp. 1617–1625, 2021.

[4] C. Li, J. Li, Y. Li, L. He, X. Fu, and J. Chen, "Fabric defect detection in textile manufacturing: a survey of the state of the art," Security and Communication Networks, vol. 2021, pp. 1–13, 2021.

[5] S. Zhao, L. Yin, J. Zhang, J. Wang, and R. Zhong, "Real-time fabric defect detection based on multi-scale convolutional neural network," IET Collaborative Intelligent Manufacturing, vol. 2, no. 4, pp. 189–196, 2020. 12

[6] Habib, T., Faisal, R., Rokonuzzaman, M., Ahmed, F. (2014). Automated fabric defect inspection: a survey of classifiers. International Journal in Foundations of Computer Science & Technology (IJFCST), 4(1), 17-25.

[7] Hanbay, K., Talu, M., Özguven, Ö. (2016). Fabric defect detection systems and methods. A systematic literature review. Optik, 127, 11960-11973.

[8] Goyal, A. (2018). Automation in fabric inspection, in Automation in Garment Manufacturing, Woodhead Publishing, 75-107.

[9] Hillel, A. B., Lerner, R., Levi, D., Raz, G. (2014). Recent progress in road and lane detection: a survey. Machine Vision and Applications, 25, 727-745.

[10] K. VINAYAN "AITEX Fabric Image Database" https://www.kaggle.com/datasets/nexuswho/aitex-fabric-image-database/data

[11] L. Cheng, J. Yi, A. Chen, and Y. Zhang, "Fabric defect detection based on separate convolutional unet," Multimedia Tools and Applications, vol. 82, no. 2, pp. 3101–3122, 2023.

[12] Y. Huang, J. Jing, and Z. Wang, "Fabric defect segmentation method based on deep learning," IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1–15, 2021.

[13] Z. Liu, J. Wang, C. Li, B. Li, and R. Yang, "Fabric defect detection using fully convolutional network with attention mechanism," in Proceedings of the 2019 8th International Conference on Computing and Pattern Recognition, 2019, pp. 134–140

[14] Y. Li, W. Zhao, and J. Pan, "Deformable patterned fabric defect detection with fisher criterion-based deep learning," IEEE Transactions on Automation Science and Engineering, vol. 14, no. 2, pp. 1256–1264, 2016.

[15] J. Jing, Z. Wang, M. Ratsch, and H. Zhang, "Mobile-unet: An effi- ¨ cient convolutional neural network for fabric defect detection," Textile Research Journal, vol. 92, no. 1-2, pp. 30–42, 2022.

[16] M. Kopaczka, M. Saggiomo, M. Guttler, K. Kielholz, and D. Merhof, ¨ "Detection and classification of faulty weft threads using both featurebased and deep convolutional machine learning methods," in Pattern Recognition Applications and Methods: 7th International Conference, ICPRAM 2018, Funchal, Madeira, Portugal, January 16-18, 2018, Revised Selected Papers 7. Springer, 2019, pp. 141–163.

[17] G. Sun, Z. Zhou, Y. Gao, Y. Xu, L. Xu, and S. Lin, "A fast fabric defect detection framework for multi-layer convolutional neural network based on histogram back-projection," IEICE TRANSACTIONS on Information and Systems, vol. 102, no. 12, pp. 2504–2514, 2019.

[18] L. Rong-qiang, L. Ming-hui, S. Jia-chen, and L. Yi-bin, "Fabric defect detection method based on improved u-net," in Journal of Physics: Conference Series, vol. 1948, no. 1. IOP Publishing, 2021, p. 012160.

[19] W. Ouyang, B. Xu, J. Hou, and X. Yuan, "Fabric defect detection using activation layer embedded convolutional neural network," IEEE Access, vol. 7, pp. 70 130–70 140, 2019.

[20] L. Shao, E. Zhang, Q. Ma, and M. Li, "Pixel-wise semisupervised fabric defect detection method combined with multitask mean teacher," IEEE Transactions on Instrumentation and Measurement, vol. 71, pp. 1–11, 2022.

[21] I. Koulali and M. T. Eskil, "Unsupervised textile defect detection using convolutional neural networks," Applied Soft Computing, vol. 113, p. 107913, 2021.

[22] J. Liu, C. Wang, H. Su, B. Du, and D. Tao, “Multistage gan for fabric defect detection,” IEEE Transactions on Image Processing, vol. 29, pp. 3388–3400, 2019.

[23] S. Niu, B. Li, X. Wang, S. He, and Y. Peng, “Defect attention template generation cyclegan for weakly supervised surface defect segmentation,” Pattern Recognition, vol. 123, p. 108396, 2022.

[24] B. Li, Y. Zou, R. Zhu, W. Yao, J. Wang, and S. Wan, “Fabric defect segmentation system based on a lightweight gan for industrial internet of things,” Wireless Communications and Mobile Computing, vol. 2022, 2022

[25] Rebhi, A., Benmhammed, I., Abid, S., Fnaiech, F. (2015). Fabric defect detection using local homogeneity analysis and neural network. Journal of Photonics, 2015, 2015.

[26] ASTM. (2016). Standard Terminology Relating to Fabric Defects. Designation: D 3990 – 12,” ASTM, West Conshohocken.

[27] Ahmed, A. (2016). A catalogue of visual textile defects. TS3B.

[28] Shorten, Connor; Khoshgoftaar, Taghi M. (2019). "A survey on Image Data Augmentation for Deep Learning". Mathematics and Computers in Simulation. springer. 6: 60.

[29] Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; Kegelmeyer, W. P. (2002-06-01). "SMOTE: Synthetic Minority Over-sampling Technique". Journal of Artificial Intelligence Research.