COVID19 HASTALIĞI İLE İLGİLİ HASTA BİLGİLERİYLE HASTANIN YOĞUN BAKIM ÜNİTESİNE YATIŞ DURUMUNUN MAKİNE ÖĞRENMESİ ALGORİTMALARIYLA TAHMİNLENMESİ

İbrahim Erkut KORAL

Haziran-2022

COVID19 HASTALIĞI İLE İLGİLİ HASTA BİLGİLERİYLE HASTANIN YOĞUN BAKIM ÜNİTESİNE YATIŞ DURUMUNUN MAKİNE ÖĞRENMESİ ALGORİTMALARIYLA TAHMİNLENMESİ

151320173025

İbrahim Erkut KORAL

DÖNEM PROJESİ

151338632 MODERN ÜRETİM TEKNOLOJİLERİ ARAŞTIRMALARI

Danışman: Dr. Öğr. Üyesi YELİZ BURUK ŞAHİN

Eskişehir Osmangazi Üniversitesi

Endüstri Mühendisliği Bölümü

Haziran – 2022

ONAY SAYFASI

151320173025 numaralı öğrencimiz İbrahim Erkut KORAL'ın MÜHENDİSLİK ÇÖZÜMLEMELERİ kapsamında hazırladığı "COVID19 HASTALIĞI İLE İLGİLİ HASTA BİLGİLERİYLE HASTANIN YOĞUN BAKIM ÜNİTESİNE YATIŞ DURUMUNUN MAKİNE ÖĞRENMESİ ALGORİTMALARITLA TAHMİNLENMESİ" başlıklı MODERN ÜRETİM TEKNOLOJİLERİ ARAŞTIRMALARI dersi dönem projesi jürimiz tarafından okunmuş ve kabul edilmiştir.

../../..

Danışman:

Dr. Öğr. Üyesi YELİZ BURUK ŞAHİN

ÖZET

COVID19 son yıllarda hayatımıza giren ve büyük bir hızla yayılarak pandemi sıfatını alan üst solunum yollarını etkileyen bir virüstür. COVID19 ile son yıllarda artan veri birikimiyle beraber makine öğrenmesi çalışmaları hem güncel bir trend hem de yeni bir çalışma alanı oluşturmaktadır. Bu çalışma kapsamında Kaggle veri bilimi platformunda bulunan COVID19 pre-condition data set kullanılarak; ham veri, veri ön işleme adımlarından geçirilip, makine öğrenmesi modelleri ile hastaneye gelen hastaların yoğun bakım ünitesine yatış durumu tahminlenmeye çalışılmıştır. Bu tahminleme sırasında üç farklı makine öğrenmesi modeli kullanılmış olup, bunlar; GBM, XGBOOST ve LightGBM'dir. Bu üç model kurulduktan sonra Super Learner işlemine tabi tutulup, meta sınıflandırıcı olarak LightGBM, temel sınıflandırıcı olarak GBM ve XGBOOST kullanılarak yaklaşık %99 başarı sağlanmıştır. Ayrıca veri setinin ön işleme aşamasından geçmemiş hali ile aynı işlemler tekrarlanılıp, yaklaşık %88 başarı yakalanarak veri ön işlemenin önemi ortaya konulmuştur.

Anahtar Kelimeler: Makine Öğrenmesi, Sınıflandırma Modelleri, Veri Ön İşleme, Karar Ağacı Modelelleri, Super Learner, Tahminleme

ABSTRACT

COVID 19 is a virus that has entered our lives in recent years and spreads rapidly and affects the upper respiratory tract, which has become a pandemic. With the increase in data accumulation in recent years with COVID19, machine learning studies create both a current trend and a new field of study. Within the scope of this study, using the COVID19 pre-condition data set on the Kaggle data science platform; The raw data was passed through the data preprocessing steps and the hospitalization intensive care unit status of the patients who came to the hospital was tried to be estimated with machine learning models. Three different machine learning models were used during this estimation, these are; GBM are XGBOOST and LightGBM. After these three models were modelled, they were subjected to the Super Learner process and approximately 99% success was achieved by using LightGBM as meta classifier, GBM and XGBOOST as basic classifier. In addition, the same operations were repeated with the data set that did not pass the pre-processing stage, and the importance of data pre-processing was demonstrated by achieving approximately 88% success.

Keywords: Machine Learning, Classification, Data Preprocessing, Decision Tree Models, Super Learner, Prediction

TEŞEKKÜR

Bu çalışmada bana bilgilerini sunup yardımcı olan ve her aşamasında motivasyon ve istikrar konusunda yanımda olan, destekleyen ve yönlendiren sayın hocam Dr. Öğr. Üyesi YELİZ BURUK ŞAHİN'e en içten teşekkürlerimi sunarım. Ayrıca bu süreçte yanımda olan arkadaşlarım Halil Derya Şenli ve Uğurcan Alparslan'a teşekkürü borç bilirim.

Aileme destekleri için teşekkür ederim.

İçindekiler

ÖZET iv

ABSTRACT v

SİMGELER VE KISALTMALAR DİZİNİ ix

1.GİRİŞ 10

1.2.HASTANEYE YENİDEN KABUL 10

2.LİTERATÜR TARAMASI 11

3.MATERYALLER VE METOTLAR 13

3.1. CRISP-DM METODOLOJISI 13

1.İşin Anlaşılması 13

2.Verinin Anlaşılması 14

3.Verinin Hazırlanması 14

4.Modelleme 14

5.Değerlendirme 14

6.Dağıtım 14

3.2. ÇALIŞMANIN YOL HARİTASI VE DENEY ORTAMI 15

3.2.METOTLAR 16

3.2.1 Karar Ağaçları 16

3.2.2 Boosting Algoritmaları 19

3.2.3 Gradient Boosting Machines 20

3.2.4 Xtreme Gradient Boosting Machines (XGBOOST) 21

3.2.5 Light Gradient Boosting Machines (LightGBM) 22

3.2.6 SUPER ÖĞRENİCİ YA DA İSTİFLEME 23

3.2.6 Hiper-parametre Tunning 25

3.2.6.1 Hiper-parametre ya da Parametre 25

3.2.7 K-Katlı Çapraz Doğrulama 26

3.2.8 Dengeli ve Dengesiz Veri Seti 26

3.2.8.1 Synthetic Minority Oversampling Technique (SMOTE) 27

4.Veri Seti Açıklaması 27

5.UYGULAMA 30

Veri Ön İşleme 31

Öznitelik Seçimi (Feature Selection) 38

Veri Setinin Dengelenmesi 39

Train, Test, Split İşlemi 40
Modelleme 40
6.DEĞERLENDİRME 42
Overfitting 45
Modelin Fazla Öğrendiğini Nasıl Anlarız? 45
Veri Ön İşleme Adımları Olmasaydı? 46
SONUÇ VE GELECEK ÇALIŞMA İÇİN ÖNERİLER 46
KAYNAKÇA 47

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar	Açılımı
GBM	Gradient Boosting Machines
XGBM	Xtreme Gradient Boosting Machines
LightGBM	Light Gradient Boosting Machines
CRISP-DM	Cross-Industry Standard Process for Data Mining

1.GİRİŞ

COVID-19'a sebep olan şiddetli akut solunum sendromu coronavirus-2 (COVID-2), 2019 aralık ayının sonlarında bir araştırma sırasında Çin'in, Wuhan şehrinde salgına sebep olmuştur. Dünya'nın genelinde vakalar çok hızlı arttığı için Dünya Sağlık Örgütü (WHO), Mart 2020'de hastalığı pandemi olarak ilan etmiştir. 2021 yılında ise vakalar eşik limitini geçtiği için kontrol altına alınamaz hale gelmiştir. (Punn vd. 2020). Virüs, canlı bir vücudun sağlıklı hücrelerine girer ve kendisini çoğaltarak vücudun organlarında kopyalar oluşturur ve sonuçta bazı sağlıklı hücrelerin ölümüne yol açar ve bu nedenle bağışıklık sistemini zayıflatır. Bu virüs ilk aşamalarda solunum sistemini etkileyerek zatürreye, organ yetmezliğine ve son aşamalara doğru ölüme neden olur. (Jamshidi vd. 2020). Hastalık, zayıf bağışıklık sistemi olan, diyabet, yüksek kan basıncı, kalp-damar sistemi hastalıkları ve solunum sistemi gibi kronik hastalığa sahip olan yaşlı bireylerde daha belirgin olmakla beraber ölüme sebebiyet vermektedir. (Yan vd. 2020).

1.2.HASTANEYE YENİDEN KABUL

Hastaneye yeniden kabul, hastane bakım kalitesinin kabul görmüş bir ölçütüdür (Mahajan SM vd. 2019). İlk taburcu olduktan sonra 30 ila 60 gün arasında belirli bir süre içinde aynı hastanede yeni yatış olarak tanımlanmaktadır (Baillie CA vd. 2013 ve Tavares MG vd. 2020). Yüksek yeniden kabul oranları, büyük olasılıkla hastaneler ve diğer sağlık merkezleri tarafından önceki kabul sırasında veya sonrasında verilen bakımın kalitesiyle ilgilidir (Goto T vd. 2019 ve Hemmrich M vd. 2019). Yeniden kabul durumunun hastanelere ve hastalara getirdiği yüksek maliyet nedeniyle, yeniden kabul durumu, bakımın kalitesini ve tahliye prosedürlerini değerlendirmede en önemli kriterlerden biridir. Tahminler gösterir ki; %60 yeniden kabul önlenebilmektedir (Wallmann R vd. 2013 ve Dharmarajan K vd. 2013). Covid19 yaygınlığı sebebiyle, birçok ülkedeki sağlık sistemleri çökmekte ve büyüyen hasta bakımı, tanı koyma ve bakım hizmetlerine yetişememektedir (Navik U vd. 2020 ve Hu Y vd. 2020). Bu durumdaki birçok hasta kabul edildikten sonra kısmen iyileşerek taburcu edilmiştir. (Donnelly JP vd. 2021). Bu sırada, hastalığın bilinmez ve agresif yapısından dolayı hastaların yeniden kabul oranı artmaya başlamıştır. (Chaudhry Z vd. 2021). Yeniden kabul, hastalara ve hastanelere ek bakım maliyeti getirmektedir. Ek olarak, bu durum

servis dağıtımı kalite göstergelerini düşürerek, ciddi komplikasyonların oluşmasını ve ölüm oranlarını arttırma yönünde eğilim göstermektedir. (Murrow JR vd. 2022). Resmi raporlara dayanarak, hastane bakım hizmetlerinden faydalanması gerekli olan Covid19 hastalarının %5'i ve bu hasta kabul raporlarından geri kabul ücretleri %2-%10 arasında değişmektedir (Naghavi S vd. 2021 ve Szente Fonseca SN. vd. 2020). Bu durumda sağlık sisteminin pandemiye karşı kapasitesini artırmak, Klinik Karar Destek Sistemleri (CDSSs) gibi teknolojik ve akıllı tabanlı çözümlere dikkat etmeyi gerektirir. (Shanbehzadeh M vd. 2021 ve Moulaei K vd. 2022). Hasta seviyesinde ulaşılabilir çok fazla veri mevcut olduğu için CDSSs'e ilgiler artmıştır. (Rojas JC vd. 2018 ve Nopour R vd. 2021). CDSSs, Covid19 hasta yeniden kabul durumuna ilişkin değerli bilgiler sunabilir. Çünkü elimizde ulaşılabilir büyük miktarda hasta verisi var ve bu veriler ulaşılabilir. (Rodriguez VA vd. 2021 ve Huang CD vd. 2020). Makine öğrenmesi algoritmaları, büyük veri setlerinde karmaşık ve esnek sınıflandırma modelleriyle veri setindeki yeni ve pratik yolları açığa çıkarabilmesiyle öne çıkmaktadır. (Rojas JC vd. 2018 ve Kalagara S vd. 2018). Makine öğrenmesi algoritmaları; riskleri değerlendirmek, riskleri görüntülemek, tahminlemek ve sağlık planlaması yapmak için geçerli ve bilime dayalı yöntemlerle tanı ve tahminleme modelleri tarafından Covid19 hastalığı ile alakalı belirsizlikleri indirger. (Chaurasia V vd. 2020 ve Ghafouri-Fard S vd. 2021). Son zamanlarda yayımlanan çalışmalar gösterir ki; hastanelik olan Covid19 hastalarının klinik çıktılarını tahminlemede birçok makine öğrenmesi metodu temel istatistik uygulamalarından daha doğru sonuçlar vermektedir. Bu modeller; hastanın hastanede kalma süresini (LOS), hastane yatak doluluğu ve devri, yoğun bakım ünitesi yeniden kabul (ICU) ve solunum entübasyonu gibi özellikleri tahminlemektedir (Shanbehzadeh M vd. 2021 ve Lorenzen SS vd. 2021). Çeşitli çalışmaları incelerken görünen o ki, Makine Öğrenmesi, artan COVID-19 vakalarını tahmin etmek için en iyi tahmin modeli gibi görünmektedir. Makine Öğrenmesi'nin regresyon ve sınıflandırma yaklaşımı, bu sorunu teşhis etmek için verilerin durumuna göre çalışmaktadır. (Phogat vd. 2021)

2.LİTERATÜR TARAMASI

COVID19 ile ilgili olarak sadece tıp bilimlerinde değil, bilimsel topluluğun tüm açılarından sürekli artan bir literatür var. Bu literatür taramasını sadece COVID19

üzerine uygulanan makine öğrenmesi algoritmaları açısından sınırlı tutmakla beraber diğer bilimsel açılardan da uygulamalar mevcut. Fernandes (2020), Atkeson (2020) ve Makridis ve Hartley (2020) COVID19'un ekonomik etkisini araştırken Wang, Zhang, Zhao, Zhang, ve Jiang (2020) COVID19 karantinasının çocuklardaki psikolojik etkisini analiz etmişlerdir. Bugüne kadar yapılan klinik çalışmalarda, COVID19 hastaları akciğer enfeksiyonuna sahipti ve bu nedenle birçok akademisyen erken otomatik tespit sistemleri için X-ray görüntülerini araştırdı. Apostolopoulos ve Mpesiana (2020), Narin, Kaya ve Pamuk (2020) ve Zhang, Xie, Li, Shen, ve Xia (2020), hastaların COVID19 olup olmadığını sınıflandırmak için akciğer X-ray görüntülerine farklı sinir ağları uyguladılar. Wang ve Wong (2020), COVID19 olan hastaları tespit etmek için göğüs X-ray görüntülerine derin evrişimsel ağ uygulaması yaptılar. Veri setlerini 13.975 göğüs röntgeni görüntüsünü içeren açık kaynaklı bir kıyaslama veri seti olarak yayınladılar. Majeed, Rashid, Ali ve Asaad (2020), X-ray görüntülerine 12 evrişimsel sinir ağları uyguladılar. COVID19 olmayan viral enfeksiyonlar, bakteriyel enfeksiyonlar ve normal X-ray görüntülerinin olduğu 2 farklı geniş çaptaki veri setlerini kullandılar. Shi ve diğerleri (2020), COVID19 ile ilgili olan görüntüsel verilere uygulanan yapay zekâ metotlarını inceleyen kapsamlı bir literatür taraması sundular. Randhawa ve diğerleri (2020) 29 COVID19 virüs dizisi içeren 5000'den fazla kendine has viral genomik dizileri analiz ettiler ve bunlara karar ağaçları yaklaşımı uyguladılar. Arentz ve diğerleri (2020) Washington State hastanesinde bulunan 21 kritik COVID19 hastasına ait hasta karakteristiklerini tartıştılar. Hastaların yaş ortalaması 70 (minimum 43, maksimum 93) ve %52 erkek hastalardan oluşmaktaydı. Bu çalışmayla ilgili kritik durumdaki hastaların özellikleri, 889/μL ortalama mutlak lenfosit sayısı, 215 ortalama trombosit sayısı 103/μL ve ortalama 515/μL beyaz kan hücresi sayısıydı. Wynants ve diğerleri (2020), akademik topluluktaki 27 çalışma ve 31 tahmin modelini incelediler ve kritik değerlendirmeler yaptılar. COVID19 olan hastalar için en önemli tahminleyiciler yaş, cinsiyet, tomografi tarama özellikleri, C reaktif proteinler, laktik dehidrojenaz ve lenfosit sayısı olarak bulundu. Tüm çalışmaların temsili olmayan kontrol grubu hasta seçimi ve overfitting nedeniyle yüksek yanlılık riski altında olduklarını belirttiler. Salman, Abu-Naser, Alajrami, Abu-Nasser, ve Alashqar (2020) 260 X-ray görüntüsüne COVID19 hastalarını tespit etmek için derin öğrenme modelleri uygularken %100 sensivity, %100 specificity, %100 accuracy, %100 pozitif tahmin, %100 negatif tahmin

skorlarına ulaştılar. Yan ve diğerleri(2020b) COVID19 hastalarını analiz ettiler ve şunları buldular; ateş ilk yaygın semptom. Bunu öksürük, halsizlik ve nefes darlığı takip etmektedir. 300'den fazla değişken kullanarak; laktik dehidrojenaz, lenfosit ve yüksek hassaslıktaki C-reaktif proteini klinik özelliklerdeki anahtar özellik olduğu sonucuna çıkmışlardır. Chen ve diğerleri (2020) COVID19'un hamilelikteki klinik karakteristiklerini analiz etmişlerdir. 9 hasta içerisinden; 7 tanesi ateş belirtisi, 4 tanesi öksürük belirtisi, 3 tanesi kas ağrısı belirtisi, 2 tanesi boğaz ağrısı belirtisi gösterdiğini gördüler. COVID19 ölüm durumunu tahminleme üzerine gelişen makine öğrenmesi modelleri ayrıca literatürde hızla büyükmektedir. Bu şekilde olan çalışmaları şu şekilde listeleyebiliriz: Chansik ve diğerleri (2020), Assaf ve diğerleri (2020), ertsimas ve diğerleri (2020), Chowdhury ve diğerleri (2020), Di ve diğerleri (2020), Ikemura ve diğerleri (2020), Laguna-Goya ve diğerleri (2020), Lalmuanawma, Hussain, ve Chhakchhuak (2020), Malki ve diğerleri (2020), Metsky, Freije, Sabeti, Myhrvold, ve Cameron (2020), Osi ve diğerleri (2020), Peng ve Nagata (2020), Randhawa ve diğerleri (2020) ve Singh et al. (2020). Analizimizde ve daha önce listelenen birçok makale gibi, farklı makine öğrenmesi modellerinin tahmin kapasiteleri açısından özellikle COVID19 veri setleri için karşılaştırılması literatürde artan bir trend olduğunu görebiliriz.

3.MATERYALLER VE METOTLAR

3.1. CRISP-DM METODOLOJISI

Veri madenciliği, çeşitli farklı beceriler ve bilgiler gerektiren yaratıcı bir süreçtir. Fakat bu süreç için standart bir işlem çerçevesi sağlamak adına bazı metodolojilerden yararlanılır. Bunlardan biri Crisp-DM metodolojisidir. Crisp-DM metodolojisi, yapılan projenin daha az maliyetli, daha hızlı, daha güvenilir, daha sürdürülebilir ve daha yönetilebilir kılar. Bu metodoloji bize yapacağımız projeyi 6 aşamada yönlendirmemizi tavsiye eder. Bunlar;

1.İşin Anlaşılması

Bu ilk aşama, iş perspektifinden proje amaçlarını ve gereksinimlerini anlamaya ve ardından bu bilgiyi bir veri madenciliği problem tanımına ve hedeflere ulaşmak için tasarlanmış bir ön proje planına dönüştürmeye odaklanır.

2.Verinin Anlaşılması

Veri anlama aşaması, ilk veri toplama ile başlar ve verilere aşina olmak, veri kalitesi sorunlarını belirlemek, veriye ilişkin ilk içgörüleri keşfetmek veya gizli bilgiler için hipotezler oluşturmak üzere ilginç alt kümeleri tespit etmek için faaliyetlerle devam eder.

3.Verinin Hazırlanması

Veri setinin ham halinden modellenecek halinin oluşturulmasına kadar olan tüm süreci kapsar. Bu aşamalar muhtemelen birden fazla kez tekrarlanabilir ve herhangi bir sırası ya da reçetesi yoktur. Bu amaç ile yapılan bazı uygulamalar şunlardır; veri setinin temizlenmesi, normalizasyon, öznitelik seçimi, yeni özniteliklerin eklenmesi.

4.Modelleme

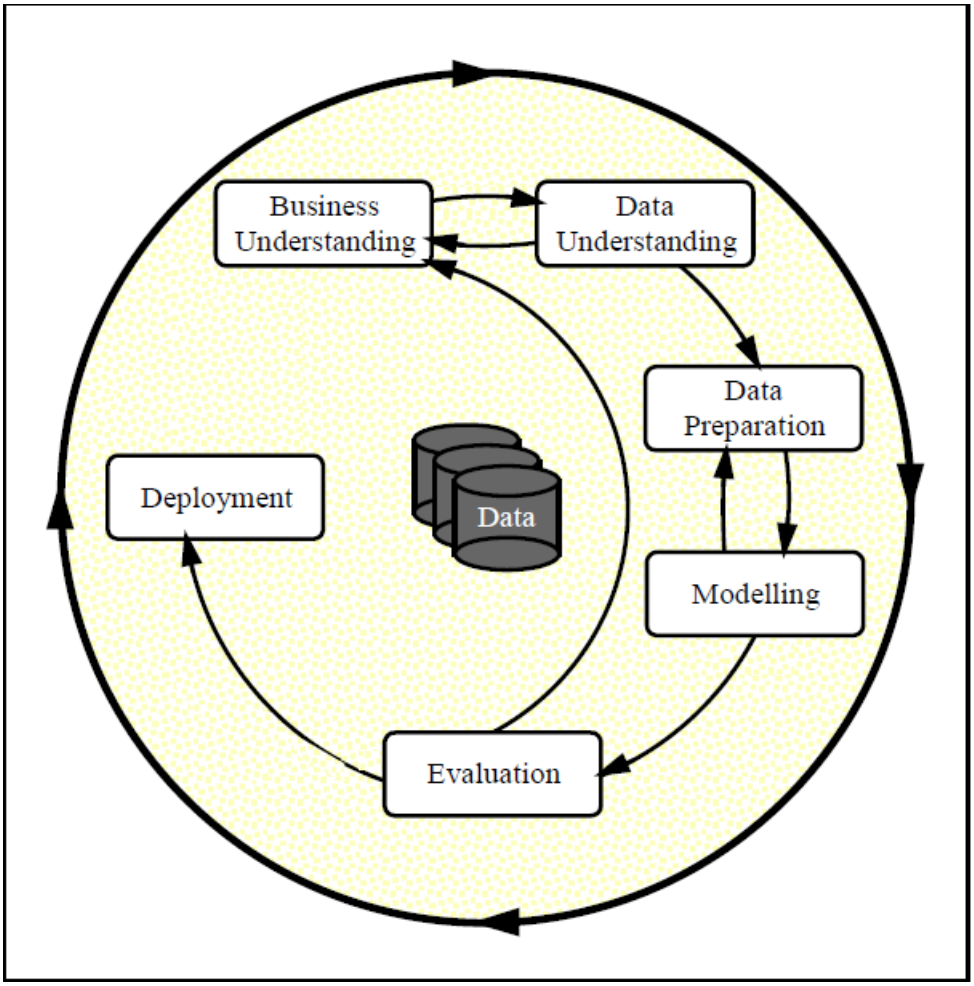
Bu aşamada makine öğrenmesi algoritmaları kullanılarak elimizdeki veri seti modellenir ve optimal parametreler için tuning işlemine alınır.

5.Değerlendirme

Modelin nihai dağıtımına geçmeden önce, modeli daha kapsamlı bir şekilde değerlendirmek ve iş hedeflerine uygun şekilde ulaştığından emin olmak için modeli oluşturmak için yürütülen adımları gözden geçirmek önemlidir. Temel amaç, yeterince dikkate alınmamış bazı önemli iş konularının olup olmadığını belirlemektir. Bu aşamanın sonunda, veri madenciliği sonuçlarının kullanımına ilişkin bir karara varılmalıdır.

6.Dağıtım

Modelin kurulması genellikle son adım olarak anlaşılsa da son adım değildir. Bu aşamada gerekli performans metrikleri ile incelenme sağlanarak modeli kullanacak birimlerin ya da kişilerin modeli anlaması sağlanır. (Wirth vd. 2000)



Şekil1: CRISP-DM Metodolojisi (Wirth vd. 2000)

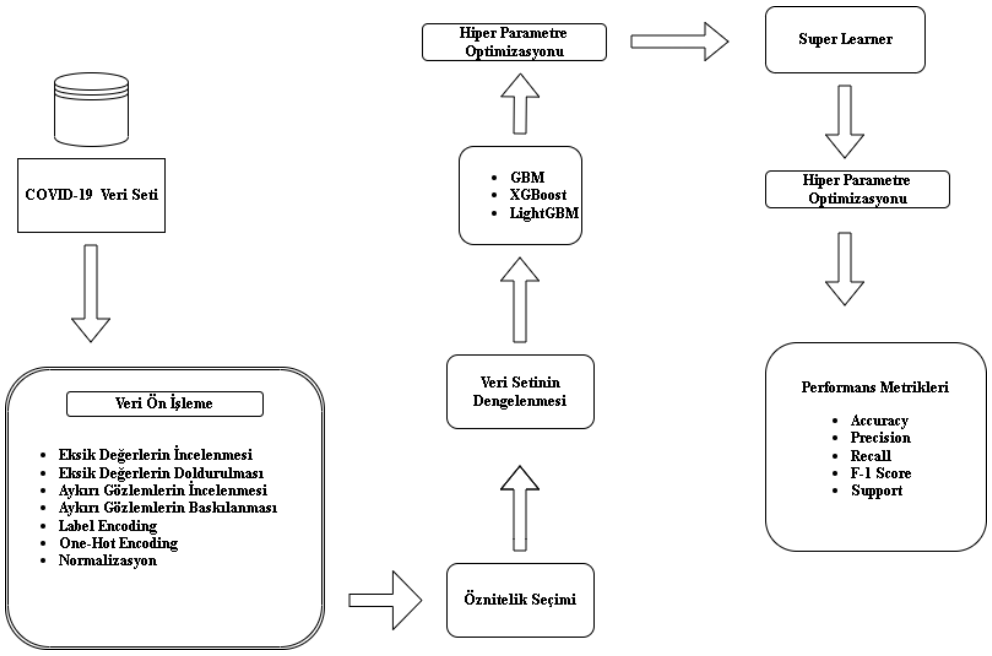
Bu metodolojiyi daha iyi anlamak için metodolojinin orijinal metni olan Reinartz & Wirth, 1995; Adriaans & Zantinge, 1996; Brachman & Anand, 1996; Fayyad et al., 1996 metninden erişilebilir.

3.2. ÇALIŞMANIN YOL HARİTASI VE DENEY ORTAMI

Bu çalışma, COVID-19 hastalarına ait bilgiler ile hastaların yoğun bakım ünitesine yatış durumunu tahminlemek üzere 2022 yılına ait en popüler yöntemlerden olan Gradient Boosting Machines (GBM), XGBoost, LightGBM ve bunların grup öğrenmesiyle oluşan Super Learner yöntemleriyle makine öğrenmesine yaklaşımı çerçevesinde tahminlenmesi üzerine kurulmuş bir çalışmadır. Çalışma aşağıdaki adımlardan oluşmaktadır;

1. COVID-19 Veri Seti
2. Veri Setinin Önişlenmesi

3. Öznitelik Seçimi
4. Veri Dengelenmesi
5. GBM, XGBM, LightGBM Uygulaması
6. GBM, XGBM, LightGBM Hiper Parametre Optimizasyonu
7. Super Learner Uygulaması
8. Super Learner Hiper Parametre Optimizasyonu
9. Performans Metriklerinin İncelenmesi



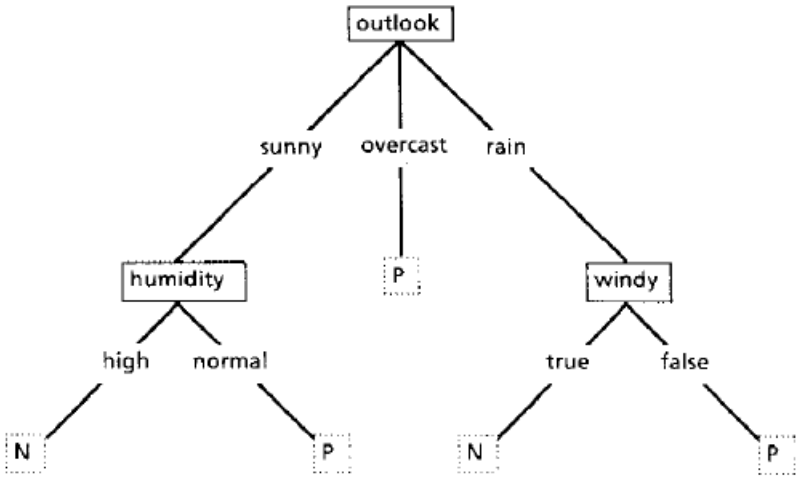
Şekil2: Çalışmanın Yol Haritası

3.2.METOTLAR

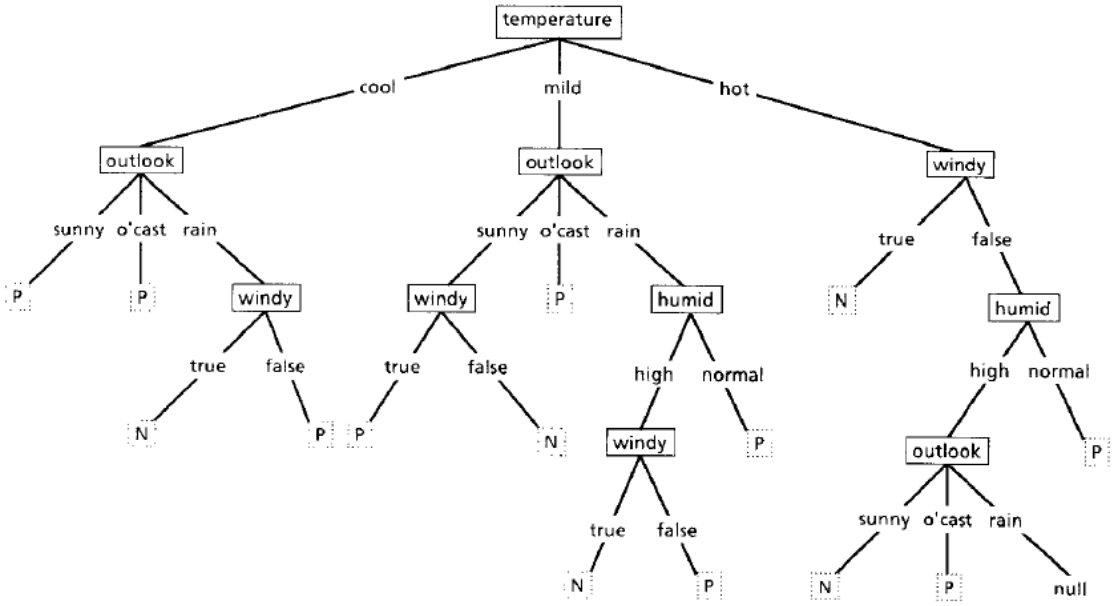
3.2.1 Karar Ağaçları

Karar ağaçları yaklaşımı, İngiliz araştırmacı William Belson tarafından 1959 yılında geliştirilmiştir. (Belson 1959). Bir karar ağacı, özyinelemeli bir yapıya sahiptir. Karar ağaçları, belirli bir veri kümesinin örneklerine dayalı olarak yukarıdan aşağıya ağaç benzeri bir model oluşturmak için kullanılır. (Hamoud vd. 2018, Nghe vd. 2007). Karar ağaçları, karar vermek için bilgi elde etmek amacıyla uygulanır. (Yadav ve Pal 2012). Ağacın yapısı; bir kök düğüm, iç düğümler ve yaprak düğümlerini (testin sonucu) içerir. (Asif vd. 2017). Bir kök düğüm, gelen bir ayrıttan oluşmaz ve bir ya da birden fazla

çıkan ayrıttan oluşur. Bir iç düğüm, bir gelen ayrıttan oluşur ve bir ya da birden fazla çıkan ayrıttan oluşur. Bir veri nesnesinin tahmin edilen son sınıfı olan yaprak düğüm, karar ağacı yapısındaki son düğümdür. Tüm iç düğümler, 2 ya da daha fazla alt düğüme sahiptirler. Karar ağaçları, 2 adım ile oluşturulur. Bunlar ağacın kurulumu ve budama kısımlarıdır. Ağaç kurulum aşaması, baştan sona doğru bir yol olarak takip edilir. Bu aşamada, ağaç yapısı, veri öğeleri aynı sınıf etiketine ait olana kadar özyinelemeli olarak bölünmeye devam eder. (Hamoud vd. 2018). Ağaç budama adımı; aşağıdan yukarıya doğru yapılır ve fazla öğrenmeyi azaltarak algoritmanın performansını geliştirmek için uygulanır. (Hamoud 2016).



Figür1: Basit Karar Ağacı (Quinlan, 1986)



Figür2: Karmaşık Karar Ağacı (Quinlan, 1986)

Karar ağacı oluşturmanın en önemli adımı, dallanma oluşturulurken hangi öznitelik değerlerinin alınacağına karar vermektir. Buna göre, entropi kurallarını içeren bilgi teorisi kullanılmaktadır. Elimizdeki veri setinin X_1, X_2,….., X_n tane sınıftan oluştuğunu ve T sınıf değeri olarak tanımlanır ise; sınıfın olasılığı;

$$P_i = (\frac{C_i}{|T|}) \quad (1)$$

Olur ve entropi sınıfı ise:

entropy(T) = $-\sum_{i=1}^{n} p_i \log_2 p_i$ (2)

Şeklinde hesaplanır.

B öznitelik değeri kullanılarak T sınıfı değerlerinin bölünmesi sonucunda elde edilecek bilgi kazancı aşağıdaki gibi hesaplanır:

gain(B,T) = entropy(T) $-\sum_{i-1}^{n} {|T_i|}/{T}\, entropy(T_i)$ (3)

Sonra, T kümesi için B özniteliğinin bölme bilgileri aşağıdaki gibidir:

splitinfo(B) = $-\sum_{i=1}^{n} \frac{|T_i|}{|T|} \log_2 \frac{|T_i|}{|T|}$ (4)

Bu aşamada; Kazanç Oranı aşağıdaki gibi gösterilmiştir: (Quinlan, 1986)

$$\text{GainRatio} = \frac{Gain(B,T)}{splitinfo(B)} \quad (5)$$

Karar Ağaçları, farklı tipten veri setlerini işlemekte ve eksik değerler ile başa çıkabilme konusunda önemli avantajlara sahiptir. Karar ağaçları aynı zamanda, son kullanıcı tarafından yorumlanabilir bir algoritmadır. (Keser ve Aghalarova 2021).

3.2.2 Boosting Algoritmaları

Boosting algoritması, Bartlett vd. tarafından 1998 yılında sunulmuştur. Bu algoritmada birçok model sırayla eğitilir. Tekrarlanan her iterasyonda; yanlış sınıflandırılan örneklerin ağırlığı arttırılırken, doğru sınıflandırılan örneklerin ağırlığı düşürülür. Boosting algoritması temel(zayıf) öğrenicilerin performansını arttırmak için kullanılan bir algoritmadır. Temele alınan herhangi bir algoritma için de boosting kullanarak performans arttırmaya gidilebilir. Boosting algoritmaları herhangi bir zayıf öğrenici ile kombinlenip güçlü öğreniciler oluşturur. Bu nedenle daha yüksek performans değerleri elde edilir. Boosting algoritması; birçok kaba genel kural bulmanın, tek ve yüksek doğrulukta bir tahmin kuralı bulmaktan daha kolay olabileceği varsayımına dayandığından, kabataslak kuralları bulmakla başlar. (Cao vd. 2010). Bu durum iteratif temel öğrenme algoritması olarak adlandırılır. Her seferinde eğitim örnekleri üzerinde farklı bir dağıtım veya ağırlık ile alt kümeleri besler. Temel (zayıf) öğrenme algoritması, yeni bir zayıf öğrenici taminleme kuralı oluşturur. Birçok iterasyondan sonra, bu algoritma tüm zayıf öğrenicileri tek bir tahmin kuralında toplayarak daha güçlü bir öğrenici oluşturmayı hedefler. Bu öğrenici, tek başına kalan zayıf öğrenicilerden daha doğru ve yüksek performanslıdır. (Keser ve Aghalarova 2021). Sözde kod ve uygulama detayları Algoritma 1'de gösterilmiştir.

Algorithm 1: Boosting Algorithm

Input: Training dataset $(x_i, y_i)_{i=1}^{N}$, number of instances N, number of iterations M
Output: Final classifier H_{final}
y_i in $\{-1, +1\}$ correct label of instance x_i in X
for m=1 to M
    construct distribution D_m on $\{1, \ldots, N\}$
    find base classifier ("rule of thumb")
        $H_m: X \rightarrow \{-1, +1\}$
    with small error on ε_m on D_m:
        $\varepsilon_m = Pr_{D_m}[h_m(x_i <> y_i)]$

Algoritma1: Boosting Algoritması Sözde Kodu (Keser ve Aghalarova 2021)

Boosting modeli, önceki modelin hatalarını indirgemeye yönelik ardışık süreçlerden oluşurken yüksek performanslı modeller oluşturmayı hedefler. Bu modeller, herhangi bir iyileşme görülmeyene kadar oluşturulmaya devam edilirler. Genellikle, Karar Ağaçları algoritmaları, sınıflandırma problemlerinde iyi sonuç verdikleri için kullanılırlar. (Keser ve Aghalarova 2021). Gradient Boosting Machines, XGBoost ve LightGBM algoritmaları da karar ağacı temelli boosting algoritmalarıdır. Gradient Boosting Machines, XGBoost ve LightGBM algoritmaları aşağıda açıklanmıştır.

3.2.3 Gradient Boosting Machines

Gradient Boosting Machines, boosting metodunun bir türü olarak 2002'de Friedman tarafından sunulmuştur. (Friedman 2002). GBM motudu, maliyet/kayıp fonksiyonunun minimize edilmesini sağlayacak modelin belirlenmesini hedefleyen optimizasyon algoritması olarak düşünülebilir. GBM algoritması iteratif olarak yeni zayıf öğrenicileri ekler ve bunu maliyet/kayıp fonksiyonunu azaltmak için yapar. (Keser ve Aghalarova 2021). GBM arkasındaki temel düşünce aşağıdaki gibi açıklanmıştır:

Eğitim verisinin; $\{§_i\}_{i=1}^{N}$, $§_i \in R^D$,

Verilerin etiketlerinin; $Y = \{†_i\}_{i=1}^{N}$, $\{†_i\} \in \{0,1\}$ bu şekilde olduğu varsayalım.

GBM, ağırlıklandırılmış fonksiyonların toplamından ek bir $F^*(x)$ yaklaşımı oluşturur. (Bentéjac vd. 2021). GBM, özel maliyet/kayıp fonksiyonlarının toplamı olan $L(†_i, F(§_i))$ fonksiyonunu minimize eden bir sınıflandırma fonksiyonu F(x) seçmeyi hedefler.

$$f_m(x) = F_{m-1}(x) + p_m h_m(\mathrm{x}) \tag{6}$$

p_m, m^{th} fonksiyonunun ağırlığıdır.

$$F^* = \underbrace{argmin}_{F} \sum_{i=1}^{N} \mathcal{L}(y_i, F(x_i)) \tag{7}$$

M.adımda, maliyet/kayıp fonksiyonu ise:

$$\mathcal{L}(y_i, F_{m-1}(x_i) + f_m(x_i)) \approx \mathcal{L}(y_i, F_{m-1}(x_i)) + g_i f_m(x_i) + \frac{1}{2} f_m(x_i)^{2|} \tag{8}$$

Şeklinde olur.

GBM algoritması hem regresyon hem de sınıflandırma modellerinde kullanılabilir. Yüksek doğruluk, hızlı eğitim ve hızlı tahmin süresi özellikleriyle ön plana çıkmaktadır. (Xiaojun Ma vd. 2018).

3.2.4 Xtreme Gradient Boosting Machines (XGBOOST)

XGBoost, GBM algoritmasının hızlı, taşınabilir ve dağıtılmış bir uygulamasıdır. (Chen ve Guestrin, 2016). Üstün sınıflandırma performansı sayesinde popüler hale gelmiştir. Aynı zamanda XGBoost, bagging ve boosting metotlarıyla birlikte çalışır. (Wandera vd. 2019). Yeni bir ağaç, her iterasyonda genel performansı yükseltmek için modele eklenir. XGBoost algoritmasında, önceki modellerde yapılan hatalar o sırada eğitilen modelin hesabına katılır. İlerleyen modeller de bu yapıda devam ettiği için çıktı optimize edilmiş olur. XGBoost algoritması fazla öğrenme durumunu eğitin aşamasında engelleyerek öğrenmede genel bir başarı sağlar. Algoritmada öğrenme sürecinde kullanılan amaç fonksiyonu, malilyet/kayıp fonksiyonu ve düzenleme teriminden oluşmaktadır. Maliyet/kayıp fonksiyonu gerçek değerler ile tahminlenen değerler arasındaki farkı hesaplamaktadır. Düzenleme terimi ise modelin karmaşıklığını kontrol eder ve fazla öğrenme durumunu engeller. XGBoost, ölçeklendirilebilir uçtan uca ağaç boosting sistemidir. Algoritma eksik değerler ve dengelenmemiş veri setleri üzerinde başarılıdır. XGBoost ve GBM algoritmaları arasındaki en büyük fark; maliyet/kayıp fonksiyonunun, amaç fonksiyonunu hesaplarken sadece maliyet/kayıp fonksiyonunun ilk türevini kullanmasıdır. (Keser ve Aghalarova 2021). XGBoost, maliyet/kayıp fonksiyonunu ikinci dereceden Taylor açılımı ile tahminler: (Liñán ve Pérez 2015).

$$f(x + \Delta x) \cong f'(x)\Delta x + \frac{1}{2}f''(x)\Delta x^2 \quad (9)$$

GBM algoritmasının iyileştirilmiş hali olan XGBoost algoritmasının birçok seçkin özelliği vardır. En büyük avantajları aşağıdaki gibidir (Xiaojun Ma vd. 2018):

1. Düzenleme adımı, overfittingi önlemek için algoritmaya entegre edilmiştir.
2. Paralel işleme, süreci hızlandırır.
3. XGBoost, kullanıcıların esnekliği artıran kullanıcı tanımlı optimizasyon hedefleri ve değerlendirme kriterleri tanımlamasına olanak tanır.
4. XGBoost, eksik değerlerle başa çıkmak için kendine has kurallara sahiptir.
5. XGBoost, karar ağacının karmaşıklığını kontrol edebilmek için özel bir budama adımlarına sahiptir.
6. XGBoost, en uygun iterasyon sayısını uygun şekilde elde etmek için çapraz doğrulama kullanır.
7. XGBoost, mevcut modellere dayalı olarak eğitime devam edebilir.

3.2.5 Light Gradient Boosting Machines (LightGBM)

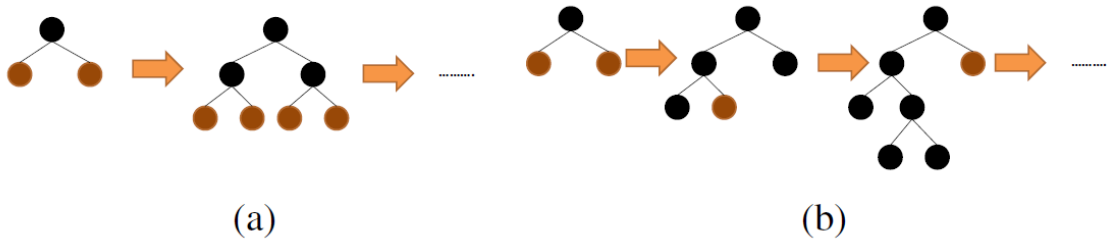
LightGBM (Light Gradient Boosting Machines) algoritması, Microsoft MSRA tarafından 2016'da tasarlanan açık kaynak promosyon çerçevesinde olan hızlı ve verimli bir GBM algoritmasıdır. (Ke vd 2017, Wandera vd, 2019). Bu algoritma; listeleme, sınıflandırma, regresyon ve başka birçok makine öğrenmesi alanında kullanılır. Aynı zamanda verimli paralel eğitim sistemini destekler. LightGBM bir GBDT türüdür. GBDT'nin büyük veri setleriyle başa çıkamaması üzerine önerilmiş bir algoritmadır. Önerilen algoritma sonrasında GBDT yapısının pratikte daha hızlı ve daha iyi bir yapıya gelmiştir. LightGBM'in alt modeli olan karar ağaçlarında, yaprak bölünmesi tarafından bölme işlemi yapılır. Bu nedenle, işleme maliyeti XGBoost'a göre nispeten daha küçüktür. Fitting olgusunu engellemek için ağacın derinliğini ve yaprak düğümdeki minimum veriyi kontrol etmemiz gerekir. LightGBM, öz değerleri birçok küçük "kova"ya bölen bir histograma dayalı olarak karar ağacı algoritmasını seçer ve ardından bu "kovalar" üzerinde depolama ve hesaplama maliyetlerini azaltabilecek bölümleri arar. Ek olarak, kategorik değişkenlerin işlenmesi LightGBM'i belli veriler için daha performanslı hale getirebilir. (Guolin, Meng vd. 2017). LightGBM, 2016'da yayınlandığı günden beri büyük veri makine öğrenmesinde yaygın olarak kullanılır. Bugün XGBoost ile makine öğrenmesinde güçlü bir araç olarak kabul edilir. Halka açık

deney verilerinde diğer boosting araçlarına göre LightGBM daha verimli, hızlı ve doğru sonuçlar verir. LightGBM, XGBoost'a göre daha hızlı, daha az hafıza gerektiren ve daha doğru çalışan bir algoritmadır. (Xiaojun Ma vd. 2018)

Ek olarak, deneyler, LightGBM'in belirli eğitim için birden fazla makine kullanarak doğrusal hızlanma sağlayabildiğini göstermektedir. Bu nedenle, LightGBM'in 5 avantajı aşağıdaki gibi verilmiştir (Xiaojun Ma vd. 2018):

1. Hızlı eğitim
2. Az hafıza tüketimi
3. İyi model hassasiyeti
4. Paralel öğrenme desteği
5. Büyük veri ile baş ederken hızlı olması

LightGBM ile ilgili daha fazla bilgi için orijinal makalesi olan Ke vd. 2017 incelenebilir.



Figür3: Seviyeli Büyüme (a) vs. Yaprak Bazında Büyüme (b) (Keser ve Aghalarova 2021)

3.2.6 SUPER ÖĞRENİCİ YA DA İSTİFLEME

Süper öğrenici ya da istifleme, bir toplu öğrenimin ikinci düzey meta öğrenicisini eğitmeyi içeren geniş bir algoritma sınıfıdır. (Ludwig vd. 2019). Super öğrenici ya da istifleme, bir üst-öğrenicinin bir temel öğreniciler koleksiyonunun çıktısı üzerinde eğitildiği bir topluluk öğrenimi prosedürüdür. (Hubbard vd. 2007). Temel öğrenicilerin çıktısı aynı zamanda seviye-1 veri olarak da adlandırılır. Bu temel öğrenicilerden alınan çıktı aynı zamanda çapraz doğrulama ile de oluşturulabilir. Orijinal eğitim veri seti genellikle seviye-0 verisi olarak anılır. (Ludwig vd. 2019). Algoritmanın sözde kodu

Algoritma2'de (LeDell 2016 ve Nykodym vd. 2016) gösterilmiş olup, konsept diyagramı ise Figür4'te (Ludwig vd. 2019) gösterilmiştir.

Algoritma2: Super Öğrenici Algoritması

1: Girdi: Veri seti D ile X örnek kadar bir set ve hedef sütunu Y

2: Çıktı: Grup Öğrenmesi Modeli

3: Grup öğrenmesi modelini kur.

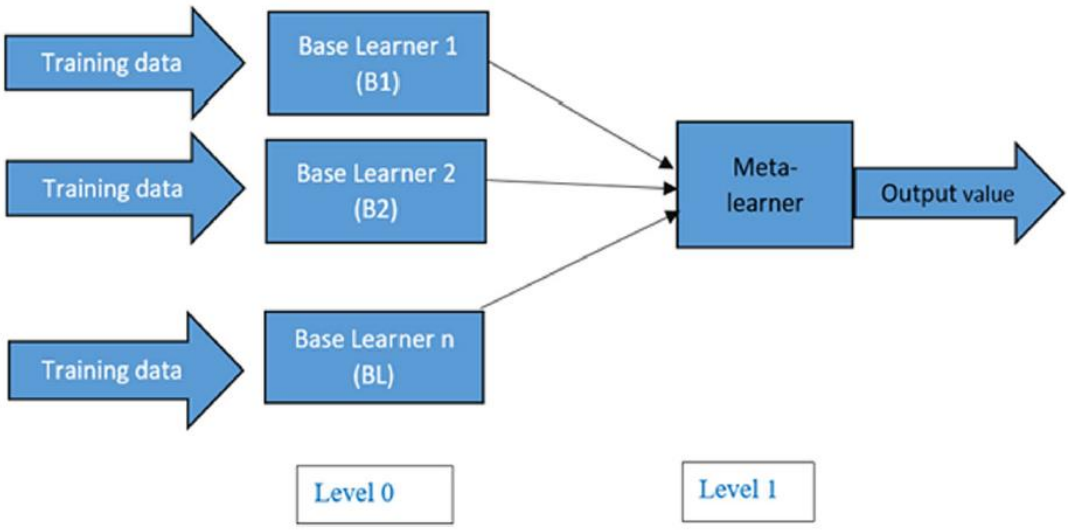
- L temel algoritma listesini belirt. (Spesifik parametreler ile)
- Meta Öğrenici algoritmasını belirt.

4: Grup öğrenmesi modelini eğit.

- L temel öğrenici setindeki tüm algoritmaları eğit.
- Her bir L öğrenicisi için k-katlı çapraz doğrulama uygula ve tüm sonuçları topla
- Toplanan N adet sonuçların tümünü bir matriste topla, Z(NXL) matrisi. Bu Z matrisi, orijinal hedef vektörü ile "seviye-1'' veri olarak adlandırılır.
- Meta öğrenim algoritmasını, seviye-1 veri (Z, Y) üzerinde eğit. Topluluk öğrenmesi modeli L temel öğrenici modeller ve meta öğrenici modelinden oluşur. Bunlar ile test setinde tahminlemer yapılabilir.

5: Yeni veri tahminle.

- Grup öğrenmesinden bir tahmin alabilmek için ilk önce temel öğrenicilerden bir tahmin alınmalıdır.
- Alınan tahminleri, meta öğrenici ile besleyerek grup öğrenmesinden tahminler alınabilir.



Figür4: Super Öğrenici (İstifleme) Yöntemi Konsept Diyagramı

3.2.6 Hiper-parametre Tunning

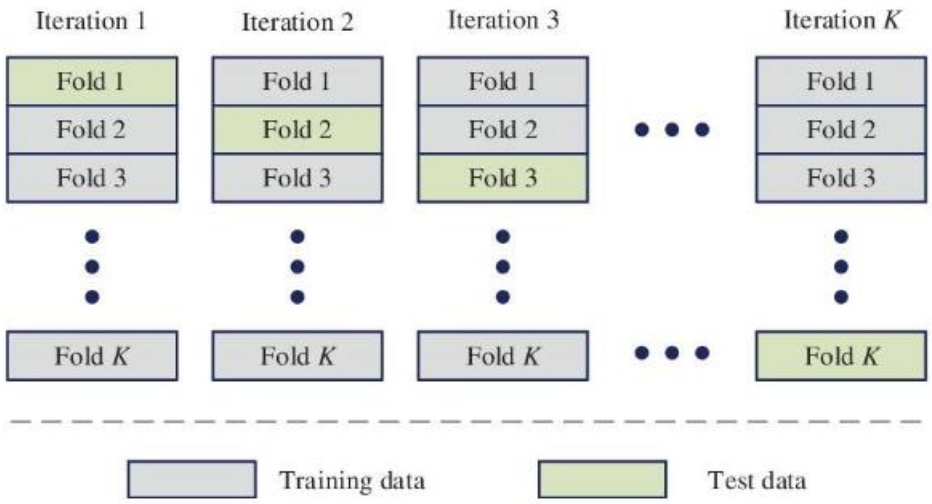
Her model için optimal (hiperparametre) ayarların belirlenmesi bir modelin tahmin gücünün sapması azaltılmış değerlendirmesi için çok önemlidir. (Schratz vd. 2019). Optimum performansları elde etmek için makine öğrenimi algoritmalarının hiper parametrelerinin ayarlanması gerekir. (Bergstra ve Bengio, 2012; Duarte ve Wainer, 2017; Hutter vd., 2011). Genellikle, parametrik modeller, optimal performansları elde etmek için ayarlama gerektirmez. Ancak bazı yarı parametrik modellerin optimize edilmesi performansı arttıracaktır. (Schratz vd. 2019).

3.2.6.1 Hiper-parametre ya da Parametre

Parametrik modeller için, "parametre" terimi genellikle, uygun bir modelin her tahmin edicisinin regresyon katsayılarına atıfta bulunmak için kullanılır. Ancak, makine öğrenimi algoritmaları için "parametre" ve "hiper parametre" terimlerinin her ikisi de "hiper parametre" anlamına gelir çünkü bu modeller için regresyon katsayıları yoktur. Ek olarak, "parametre" terimi genellikle programlamada bir fonksiyonun argümanına atıfta bulunmak için kullanılır. Bir modelin en iyi performansı isteniyorsa, hiper parametreler manuel olarak ayarlanamaz. En iyi hiper parametreler ayarlanmak isteniyorsa, otomatik optimizasyon gereklidir. Bu optimizasyonlar; bu bitirme tezinde kullanıldığı gibi “Rastgele Arama’’ (Random SearchCV) ya da Grid SearchCV ile yapılabilir. (Kuhn ve Johnson, 2013).

3.2.7 K-Katlı Çapraz Doğrulama

K-katlı çapraz doğrulamada, veriler ilk önce eşit (ya da neredeyse eşit) boyutlu bölümlere ya da katlara bölünür. Ardından, her yinelemede verilerin farklı bir katının doğrulama için tutulacağı ve kalan k-1 katlarının öğrenme için kullanılacağı şekilde k eğitim ve doğrulama iterasyonları gerçekleştirilir. Veriler genellikle k kata bölünmeden önce katmanlara ayrılır. Katmanlandırma, her katın bütünün iyi bir temsilcisi olmasını sağlamak için verileri yeniden düzenleme sürecidir. Örneğin, her sınıfın verilerin %50'sini oluşturduğu bir ikili sınıflandırma probleminde, verileri her kattaki her sınıf örneklerin yaklaşık yarısını oluşturacak şekilde düzenlemek en iyisidir. (Refaeilzadeh vd. 2009).



Figür5: K-Katlı Çapraz Doğrulama (Wayahdi vd. 2020)

3.2.8 Dengeli ve Dengesiz Veri Seti

Genel olarak, denge ve dengesizlik sınıfları, veri kümelerinin iki temsilidir. Genellikle, dolandırıcılık tespiti, hastalıkların yaygınlığı, kredi puanlaması veya tıbbi teşhis gibi birçok uygulamada gerçek dünya verileri dengesizdir. Dengesiz veri seti, denetimli bir öğrenme sorunudur ve veri bilimi topluluğunda çok popülerdir. (Alghamdi vd. 2017). Sınıf dengesizliği sorunu, çoğunluk sınıf sayısı ile azınlık sınıf sayısı arasında büyük bir fark olduğunda ve çoğunlukla ikili (binary) değerlere sahip sınıflarda ortaya çıkar. (Batista GE vd. 2004 Menardi G, vd. 2014). Bu eşitsizlik durumu, makine öğrenmesi algoritmaları performanslarında çok büyük negatif etki yaratmaktadır. (Ganganwar V. 2012). Genellikle, bu yanlış sınıflandırmaya yol açacaktır ve model çoğunluk sınıfı için

sapmayı azaltmadığı için tahmin sonucu ya fazla takılmıştır ya da çok az sayıda pozitif sınıf örneği nedeniyle düşük performans gösterecektir. (He H vd. 2009). Pratikte, birkaç çalışma, dengeli verilere sahip olarak daha iyi tahmin performansının elde edilebileceğini göstermiştir; bu nedenle, tahmin modellerinin performansını iyileştirmek için bu sorunu çözmek için makine öğreniminde bir dizi iyi bilinen yöntem geliştirilmiş ve kullanılmıştır. (Poolsawad N vd. 2014). Bu yöntemler "Örnekleme Metotları'' olarak adlandırılır. (Poolsawad N vd. 2014, Wang J vd. 2006, Garcia V vd. 2006, Jack CR vd. 2008). Bu yöntemlerin ana konsepti, orijinal veri kümesi hedef sınıf değerlerini etiket sınıfındaki dağılıma eşit olacak şekilde değiştirmektir. Eksik örnekleme ve aşırı örnekleme yöntemleri birçok biçimde uygulanmaktadır. (Lusa L vd. 2015).

3.2.8.1 Synthetic Minority Oversampling Technique (SMOTE)

SMOTE tekniği, tıpta giderek daha fazla kullanılan dengesiz yüksek boyutlu verilerle makine öğreniminde güçlü olduğu gösterilen ve yaygın olarak kullanılan bir tür aşırı örnekleme yöntemidir. (Lusa L vd. 2015). SMOTE tekniği, örnek sayısını artırmak için azınlık sınıfı örneğine katılan en yakın komşulardan azınlık sınıfının rastgele yeni örneklerini üretir. Bu örnekler, orijinal veri kümesinin özelliklerine göre oluşturulur, böylece azınlık sınıfının orijinal örneklerine benzer hale gelirler. (Chawla NV. 2005).

4.Veri Seti Açıklaması

Kullanılan veri seti Kaggle platformu üzerinde bulunan "COVID-19 patient pre-condition dataset" isimli veri setidir. Veri setiyle ilgili tüm süreçler "Python" kodlama dili üzerinden yürütülmüş olup, dağıtıcı olarak "Anaconda" kullanılmıştır. Kulanılan bilgisayarın özellikleri ise Intel i5 10300H CPU ve 8 GB RAM'dir.Veri seti Meksika'da bulunan bir hastanede hastalar ile ilişkili bazı bilgiler içermektedir. Bu bilgilerin detayına ilerleyen bölümlerde değinilecektir. Veri seti içerisinde toplamda 563201 adet veri vardır ve veri seti 23 adet sütuna sahiptir. Veri setine ve daha fazla açıklamaya bu linkten erişim sağlanabilir:

https://www.kaggle.com/datasets/tanmoyx/covid19-patient-precondition-dataset?select=covid.csv

Verinin orijinal haline ise aşağıdaki linkten erişim sağlanabilir:

https://www.gob.mx/salud/documentos/datos-abiertos-152127

Tablo 1

Değişkenler, Tanımları, Tipi ve Aralığı

Değişkenler	Tanımı	Tipi ve Aralığı
id	hastanın numarası	Nominal, her hasta için verilmiş bir sayı öbeği
sex	hastanın cinsiyeti	Kategorik, Kadın 1, Erkek 2
patient_type	hastanın tipi	Kategorik, hastanede içi 2, hastane dışı 1
entry_date	hastanın hastaneye giriş tarihi	Nominal, her hasta için verilmiş bir tarih
date_symptoms	hastada semptomların görüldüğü tarih	Nominal, her hasta için verilmiş bir tarih
date_died	hastanın öldüğü tarih	Nominal, her hasta için verilmiş bir tarih, 9999-99-99 hastanın kurtarıldığı anlamına gelir.
intubed	hasta için entübasyon gerekiyorsa	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
pneumonia	hasta için zatüre tanısı	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
age	hastanın yaşı	Numerik, 0-120 arasında değerler alır
pregnancy	hastanın hamilelik durumu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
diabetes	hasta için diyabet tanısı	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99

copd	hasta için KOAH tanısı	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
asthma	hasta için astım tanısı	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
inmsupr	hastanın zayıf bağışıklık sistemi olması durumu	Kategorik, (Evet 1, Hayır 2, veri eksik ise 97,98,99)
hypertension	hastanın hipertansiyona sahip olması durumu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
other_disease	hastanın başka hastalıklara sahip olması durumu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
cardiovascular	hastanın kardiyovasküler bir hastalığa sahip olması durumu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
obesity	hastanın obez olması durumu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
renal_chronic	hastanın kronik böbrek yetmezliği olması durumu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
tobacco	hastanın tütün ürünleri kullanması durumu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
contact_other_covid	hastanın temaslı olma duurmu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99
covid_res	hastanın covid test sonucu	Kategorik, Pozitif 1, Negatif 2, Sonuç bekliyor 3
icu	Hastanın yoğun bakım ünitesine kaldırılması durumu	Kategorik, Evet 1, Hayır 2, veri eksik ise 97,98,99

Tablo1'den de görüldüğü gibi değişkenlerimizin çoğunluğu kategorik değişkenlerden oluşmaktadır ve eksik veri ile ilgili bilgiler içeren değişkenler için eksik verilere sahibizdir. Fakat bu eksik veriler veri setinin orijinal hali tarafından görüntülenememektedir. Çünkü Python üzerinden kullanacağımız kod verileri "NaN"

tipindeyse eğer eksik veri olarak ele almaktadır. Bunun için veri setinde “97,98,99” olarak verilen eksik veri yapısı NaN formatına dönüştürülmüş olup, daha genel bir görünüm elde edilmesi için tüm kategorik değişkenler sayısal ifadelerden yazılı ifadelere dönüştürülmüştür. Örneğin cinsiyet değişkeni için 1, kadın cinsiyetini ifade ettiği için 1 içeren tüm örnekler “Female” ile 2, erkek cinsiyetini ifade ettiği için 2 içeren tüm örnekler “Male” kelimeleriyle değiştirilmiştir. Tüm dönüşümler yapıldıktan sonra veri setinin ilk beş örneğinin görüldüğü yapı aşağıdaki gibidir.

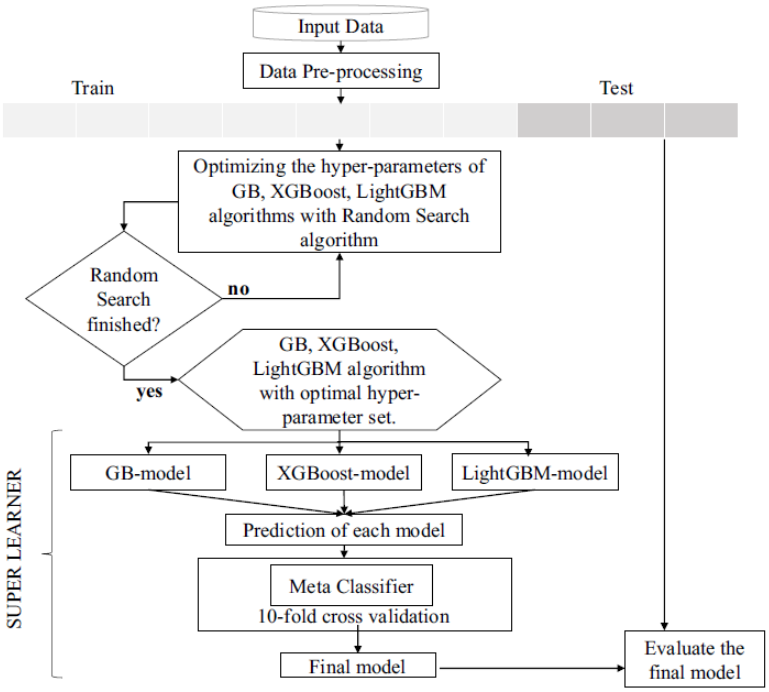
	sex	patient_type	entry_date	date_symptoms	date_died	age	covid_res	intubed	pneumonia	pregnancy	...	asthma	inmsupr	hypertension	other_disease	card
0	Male	Outpatient	04-05-2020	02-05-2020	9999-99-99	27	Positive	NaN	No	NaN	...	No	No	No	No	
1	Male	Outpatient	19-03-2020	17-03-2020	9999-99-99	24	Positive	NaN	No	NaN	...	No	No	No	No	
2	Female	Inpatient	06-04-2020	01-04-2020	9999-99-99	54	Positive	No	No	No	...	No	No	No	No	
3	Male	Inpatient	17-04-2020	10-04-2020	9999-99-99	30	Positive	No	Yes	NaN	...	No	No	No	No	
4	Female	Inpatient	13-04-2020	13-04-2020	22-04-2020	60	Positive	No	No	No	...	No	No	Yes	No	

Görüntü1: Veri Seti Düzenlemesi Sonrası

5.UYGULAMA

Bu kısım içinde tahminlenecek öznitelik seçilmiş ve bu değişken “icu” değişkeni olarak kabul edilmiştir. Bu doğrultuda bu veri setinden yararlanarak hastanın yoğun bakım ünitesine yatıp yatmadığı tahminlenmek üzere süreç geliştirilmeye başlanmıştır.

Hedef değişkeninin seçilmesinin ardından; veri ön işleme, GBM, XGBOOST, LightGBM modellerinin uygulanması, ilgili modeller için hiper parametre tuning, ve bu modellerin Super Learner içersinde kullanılması şeklinde ilerleyen bir yapı üzerine karar verilmiş olup yapının akış diyagramı aşağıdaki gibidir. (Keser ve Aghalarova 2021)



Şekil3: Çalışma Akış Diyagramı (Keser ve Aghalarova 2021)

Veri Ön İşleme

Veri ön işleme adımında, Python içerisindeki "Pandas", "Sci-kit Learn" ve "Label Encoder" kütüphaneleri kullanılmıştır. İlk aşama olarak veri setinin içindeki gereksiz değişkenleri çıkartmak yönüne gidildi ve bu aşamada gereksiz değişkenler olarak tarih içeren değişkenlerimiz "entry_date", "date_symptoms", "date_died" değişkenleri veri setinden çıkartıldı.

```
df.drop(["entry_date"], axis = 1, inplace = True)
```

```
df.drop(["date_symptoms"], axis = 1, inplace = True)
```

```
df.drop(["date_died"], axis = 1, inplace = True)
```

Görüntü2: Tarih İçeren Değişkenlerin Çıkarılması

İkinci aşama olarak eksik verilerin analiz edilmesiyle başlanmış olup, veri setinin eksik gözlemleri aşağıdaki görüntüdedir.

```
df.isnull().sum()
```

```
sex                        0
patient_type               0
age                        0
covid_res                  0
intubed                444813
pneumonia                  11
pregnancy              288699
diabetes                 1981
copd                     1749
asthma                   1752
inmsupr                  1980
hypertension             1824
other_disease            2598
cardiovascular           1822
obesity                  1781
renal_chronic            1792
tobacco                  1907
contact_other_covid    175031
icu                    444814
dtype: int64
```

Görüntü3: Preprocessing Öncesi Eksik Gözlemler

Görüntü2'den de görüleceği üzere veri seti içerisindeki cinsiyet, hasta tipi, yaş ve covid testi değişkenleri için eksik gözlem olmamakla beraber diğer değişkenler için özellikle "icu" yani hastanın yoğun bakıma alınıp alınmamasıyla ilgili değişkenimizde çok fazla sayıda eksik gözlem mevcuttur.

Bu değişkenlerin baskılanması ve aynı zamanda veri kaybetmeden bunu yapabilmek için veri doldurma yöntemlerine gidilmiş ve bu yöntemler arasından en kolay ve erişilebilir olan kategorik değişkenler için mod değeri ile doldurma kullanılırken, numerik değişkenler için gerektiğinde ortalama ile doldurma kullanılmıştır. (L Peng vd. 2005).

```
df["intubed"].mode()[0]
```

```
'No'
```

```
df["intubed"].fillna(df["intubed"].mode()[0], inplace = True)
```

Görüntü4: "intubed" Değişkeninin Modu ve Mod ile Doldurulması

```
df["pneumonia"].mode()[0]
```

```
'No'
```

```
df["pneumonia"].fillna(df["pneumonia"].mode()[0], inplace = True)
```

Görüntü5: “pneumonia” Değişkeninin Modu ve Mod ile Doldurulması

```
df["pregnancy"].mode()[0]
```

```
'No'
```

```
df["pregnancy"].fillna(df["pregnancy"].mode()[0], inplace = True)
```

Görüntü6: “pregnancy” Değişkeninin Modu ve Mod ile Doldurulması

```
df["diabetes"].mode()[0]
```

```
'No'
```

```
df["diabetes"].fillna(df["diabetes"].mode()[0], inplace = True)
```

Görüntü7: “diabetes” Değişkeninin Modu ve Mod ile Doldurulması

```
df["copd"].mode()[0]
```

```
'No'
```

```
df["copd"].fillna(df["copd"].mode()[0], inplace = True)
```

Görüntü8: “copd” Değişkeninin Modu ve Mod ile Doldurulması

```
df["asthma"].mode()[0]
```

```
'No'
```

```
df["asthma"].fillna(df["asthma"].mode()[0], inplace = True)
```

Görüntü9: “asthma” Değişkeninin Modu ve Mod ile Doldurulması

```
df["inmsupr"].mode()[0]
```

```
'No'
```

```
df["inmsupr"].fillna(df["inmsupr"].mode()[0], inplace = True)
```

Görüntü10: “inmsupr” Değişkeninin Modu ve Mod ile Doldurulması

```
df["hypertension"].mode()[0]
```

```
'No'
```

```
df["hypertension"].fillna(df["hypertension"].mode()[0], inplace = True)
```

Görüntü11: “hypertension” Değişkeninin Modu ve Mod ile Doldurulması

```
df["other_disease"].mode()[0]
```

```
'No'
```

```
df["other_disease"].fillna(df["other_disease"].mode()[0], inplace = True)
```

Görüntü12: “other_disease” Değişkeninin Modu ve Mod ile Doldurulması

```
df["cardiovascular"].mode()[0]
```

```
'No'
```

```
df["cardiovascular"].fillna(df["cardiovascular"].mode()[0], inplace = True)
```

Görüntü13: “cardiovascular” Değişkeninin Modu ve Mod ile Doldurulması

```
df["obesity"].mode()[0]
```

```
'No'
```

```
df["obesity"].fillna(df["obesity"].mode()[0], inplace = True)
```

Görüntü14: “obesity” Değişkeninin Modu ve Mod ile Doldurulması

```
df["renal_chronic"].mode()[0]
```

```
'No'
```

```
df["renal_chronic"].fillna(df["renal_chronic"].mode()[0], inplace = True)
```

Görüntü15: “renal_chronic” Değişkeninin Modu ve Mod ile Doldurulması

```
df["tobacco"].mode()[0]
```

```
'No'
```

```
df["tobacco"].fillna(df["tobacco"].mode()[0], inplace = True)
```

Görüntü16: “tobacco” Değişkeninin Modu ve Mod ile Doldurulması

```
df["contact_other_covid"].mode()[0]
```

```
'Yes'
```

```
df["contact_other_covid"].fillna(df["contact_other_covid"].mode()[0], inplace = True)
```

Görüntü17: “contact_other_covid” Değişkeninin Modu ve Mod ile Doldurulması

```
df["icu"].mode()[0]
```

```
'No'
```

```
df["icu"].fillna(df["icu"].mode()[0], inplace = True)
```

Görüntü18: “icu” Değişkeninin Modu ve Mod ile Doldurulması

Modu görüntülenen ve mod ile doldurulan değişkenlerin doldurulma işleminden sonraki değişkenlerin eksik gözlemlerinin görüntüsü aşağıdaki gibidir.

```
df.isnull().sum()
```

```
sex                  0
patient_type         0
age                  0
covid_res            0
intubed              0
pneumonia            0
pregnancy            0
diabetes             0
copd                 0
asthma               0
inmsupr              0
hypertension         0
other_disease        0
cardiovascular       0
obesity              0
renal_chronic        0
tobacco              0
contact_other_covid  0
icu                  0
dtype: int64
```

Görüntü19: Preprocessing Sonrası Eksik Gözlemler

Veri setinin içinde bulunan eksik gözlemler baskılandıktan sonra, makine öğrenmesi algoritmalarının çalışmasını sağlayacak format olan 0-1 formatına veri setinin dönüştürülmesi gereklidir. Bunun için binary sınıflı değişkenler için Label Encoder kullanılmış olup, üç veya daha fazla sınıflı değişkenler için One-Hot Encoding işlemi kullanılmıştır. Label Encoder kullanılabilmesi için, ilk önce kütüphaneden ilgili fonksiyon çağırılmalıdır.

```
from sklearn.preprocessing import LabelEncoder
```

Görüntü20: Label Encoder Fonksiyonun Çağırılması

Bu işlemin ardından, binary sınıflı değişkenler belli bir grup altında toplanmıştır ve döngü kullanımı ile ilgili fonksiyon gruba uygulanmıştır.

```
binary_cat_columns = ['sex', 'patient_type', 'intubed', 'pneumonia', 'pregnancy', 'diabetes', 'copd', 'asthma', 'inmsupr', 'hypertension',
                      'other_disease', 'cardiovascular', 'obesity', 'renal_chronic', 'tobacco', 'contact_other_covid', 'icu']

lbe = LabelEncoder()

for col in binary_cat_columns:
    df[col] = lbe.fit_transform(df[col])
```

Görüntü21: Binary Sınıflı Değişkenlerin Encoding İşlemi

Veri setinde bir tane üç ya da daha fazla sınıfa sahip bir değişken bulunmaktadır fakat döngü metodunun daha iyi anlaşılması için tekrardan aynı döngü kullanımı ile One-Hot Encoding işlemi yapılmıştır.

```
nominal_cat_columns = ['covid_res']

for col in nominal_cat_columns:
    df= pd.get_dummies(df, columns = [col], drop_first = True)
```

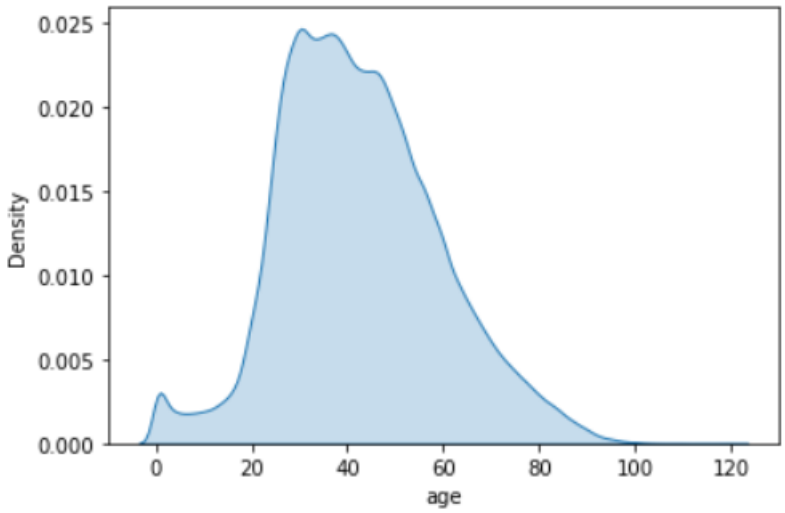
Görüntü22: One-Hot Encoding İşlemi

Veri setinin encoding işlemleri ardından görünümü aşağıdaki gibidir. Burada dikkat edilmesi gereken önemli bir nokta olarak, “age” değişkeninin 0-1 formatında olmadığıdır. Eğer veri seti böyle bırakılır ise makine öğrenmesi algoritmaları bu değişken sebebiyle yanlı öğrenecektir.

	sex	patient_type	age	intubed	pneumonia	pregnancy	diabetes	copd	asthma	inmsupr	hypertension	other_disease	cardiovascular	obesity	renal_chronic	tobacco
0	1	1	27	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	24	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	54	0	0	0	0	0	0	0	0	0	0	1	0	0
3	1	0	30	0	1	0	0	0	0	0	0	0	0	0	0	0
4	0	0	60	0	0	0	1	0	0	0	1	0	1	0	0	0

Görüntü23: Encoding İşlemleri Sonrasında Veri Setinin Bir Parçasının Görüntüsü

Değişkenleri anlama konusunda belli işlemler gereklidir. Fakat elimizdeki bir değişken haricindeki tüm değişkenler kategorik değişkenlerdir. Tek numerik değişken olan “age” değişkeninin dağılımını incelemek gerekir.

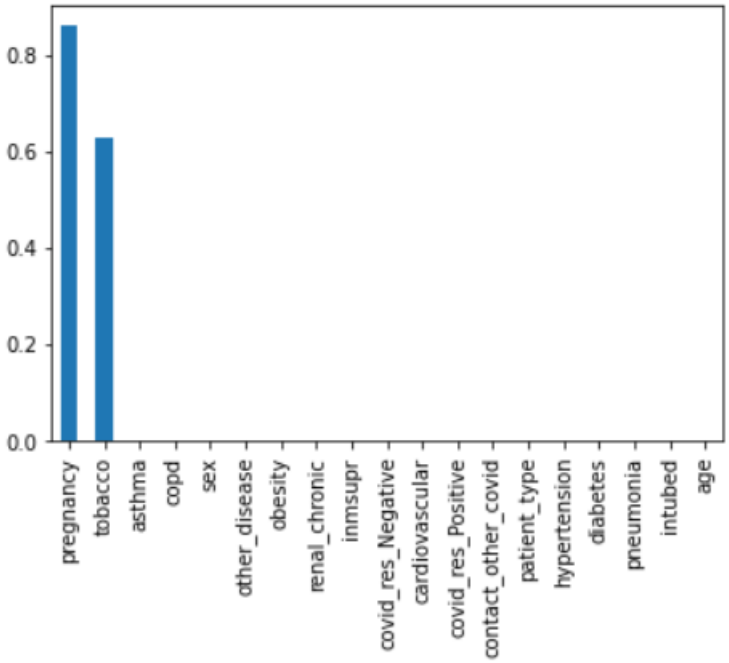


Görüntü24: “age” Değişkeninin Dağılım Tablosu

Görüntü24’ten de görüleceği üzere; “age” değişkeni normal dağılıma benzemektedir. Elimizdeki gözlemler ise 0 yaştan 120 yaşına kadar bir aralıktadır.

Öznitelik Seçimi (Feature Selection)

Feature Selection adımı için, ki-kare testi kullanılmıştır. Bunun için ki-kare ile ilgili fonksiyonu “sklearn” kütüphanesinden çekmemiz gereklidir. Veri setinde bulunan değişkenlerin ki-kare skorları aşağıdaki gibidir.



Görüntü25: Değişkenlerin Ki-Kare Skorları

Bulunan p değerlerinden 0.05'ten büyük olan "pregnancy" ve "tobacco" değişkenleri hedef değişken olan "icu" değişkeni ile anlamlı bir farklılıkta bulunmadığı için veri setinden çıkartılır.

Veri Setinin Dengelenmesi

Veri setindeki hedef değişken olan "icu" değişkeni yani hastanın yoğun bakıma yatıp yatmaması durumu dengesiz bir haldedir. Bu dengesiz hal aşağıdaki gibi görüntülenebilir.

Görüntü26: Dengesiz Hedef Değişkeni

Daha sayısal bir görünüm sağlamak için hedef değişkenin içindeki sınıfların örnek sayıları sayılabilir.

```
df.icu.value_counts()
```

```
0    556490
1     10112
Name: icu, dtype: int64
```

Görüntü27: "icu" Değişkeni için Sınıf Örnek Sayısı

Bu dengesizliğin giderilmesi sonucunda makine öğrenmesi modeli hedef değişkendeki her sınıfı aynı derecede öğrenecek ve tek taraflı tahminlemeye meyletmeyecektir.

Bu dengeleme işlemi SMOTE-N ile sağlanmış olup, dengelendikten sonraki veri seti hali aşağıdaki gibidir.

```
s_df.icu.value_counts()
```

```
0    556490
1    556490
Name: icu, dtype: int64
```

Görüntü28: Dengeli Hedef Değişken

Bu işlemin ardından "age" değişkenini için normalizasyon işlemi uygulanır ve veri ön işleme adımı sonlandırılır.

Train, Test, Split İşlemi

Veri seti, makine öğrenmesi modelleri uygulanmadan önce modelin eğitilmesi için train, modelin test edilmesi için ise test setlerine ayrılır. Bu çalışma kapsamında veri setinin %30 kısmı test için ayrılmış olup, %70 kısmı ise train için ayrılmıştır.

```
y = s_df["icu"]
X = s_df.drop(['icu'], axis=1)
X = pd.DataFrame(X)
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.30,
                                                    random_state=42)
```

Görüntü29: Veri Setinin Train ve Test Olarak Ayrılması

Modelleme

Train ve Test setine ayrılan veri seti makine öğrenmesi modelleri için hazırdır. Bu aşamada kullanılan makine öğrenmesi modelleri aşağıdaki gibidir:

1. GBM
2. XGBOOST
3. LightGBM

Ardından bu modeller için optimal parametrelerin seçilmesi kapsamında parametre optimizasyonu yapılır. Bu optimizasyon için RandomSearchCV kullanılmış olup, aratılan parametreler aşağıdaki gibidir.

Tablo2

Modeller için aranan hiper parametre aralıkları

Modeller	Hiper parametreler	Aralıklar
GBM	max_depth	[2, 3, 4, 5, 9, 10, 15, 20, 25, 35]
	min_samples_leaf	[1, 3, 2, 4, 5]
	min_samples_split	[2, 3, 4, 5, 10]
XGBOOST	max_depth	[2, 3, 4, 5, 9, 10, 15, 20, 25, 35]
	n_estimators	[50, 70, 100, 150, 200, 250, 500]
	gamma	[0.5, 1, 1.5, 2, 5]
LightGBM	max_depth	[2, 3, 4, 5, 9, 10, 15, 20, 25, 35]
	num_leaves	[2, 4, 6, 8, 10, 16, 32]
	n_estimators	[50, 70, 100, 150, 200, 250, 500]
	learning_rate	[0.01, 0.05, 0.1, 0.5, 1]
	subsample	[0.6, 0.8, 1.0]

Tablo2'de bulunan aralıklar ile aratılan optimal hiper parametre sonuçları aşağıdaki gibidir. Uygulanan yöntem olarak kullanılan RandomSearchCV çok detaylı bir kapsamda arama yapmamaktadır. Hem işlem süresini kısaltmak için hem de performansı düşük sistemlerde kullanım kolaylığı sağladığı için kullanılmıştır. Veri seti tanıtımında verilen bilgi dahilinde i5 10300H ve 8 GB dahilinde olan sistem için daha kapsamlı olan GridSearchCV yöntemi kullanılmamıştır. Çünkü hem işlem süresi çok uzun sürmektedir hem de performans olarak bulunan sistem üzerine çok baskı yaptığı için bu yönteme geçilmiştir.

Tablo3

Optimal hiper parametreler

Modeller	Hiper parametreler	Değer
GBM	max_depth	3
	min_samples_leaf	1
	min_samples_split	3
XGBOOST	max_depth	25
	n_estimators	200
	gamma	5
LightGBM	max_depth	10
	num_leaves	32
	n_estimators	500
	learning_rate	0.1
	subsample	0.6

Optimal sonucu bulunan hiper parametreler Super Learner (Stacking) uygulaması ile işlenmeye devam etmektedir. Super Learner uygulaması için meta sınıflandırıcı olarak LightGBM seçilmiştir. Hem işlem hızı kolaylığı hem de yeni nesil bir algoritma olması sebebiyle meta sınıflandırıcı olarak konumlandırılmıştır. Temel (base) sınıflandırıcılar ise XGBOOST ve GBM olarak seçilmiştir. Bu iki model hem işlem hızı olarak hem de LightGBM modeline göre daha eski nesil olmaları sebebiyle temel sınıflandırıcı olarak kullanılmıştır.

6.DEĞERLENDİRME

Yapılan çalışmanın değerlendirilmesi için confusion matrix kullanılmıştır. Confusion matrisi, algoritmanın verimliliğini belirlemek ve sınıflandırıcıların farklı sınıfların demetlerini ne kadar iyi tanıdığını analiz etmek için kullanılan ana araçlardan biridir. (Lagman vd. 2020). Doğru ve yanlış tahminlerin sayısı, her sınıfın sayı değerleriyle birlikte özetlenir. Bu kapsamda 4 farklı metrik incelenmektedir. Bunlar;

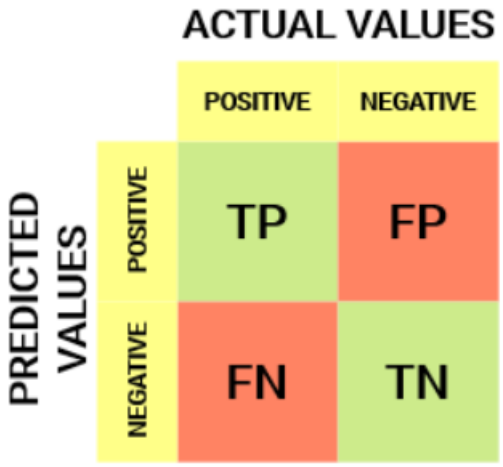
True Positives (TP): Sınıflandırıcının hasta olarak sınıflandırdığı örnekler.

True Negatives (TN): Sınıflandırıcının sağlıklı olarak sınıflandırdığı örnekler.

False Positives (FP): Sınıflandırıcının hasta olanları sağlıklı olarak sınıflandırdığı örnekler. Ayrıca Tip1 hata olarak bilinir.

False Negatives (FN): Sınıflandırıcının sağlıklı olanları hasta olarak sınıflandırdığı örnekler. Ayrıca Tip2 hata olarak bilinir.

Bu çalışmada 2x2 confusion matrisi kullanılmış olup, pilot görsel aşağıdaki gibidir.



Şekil4: Confusion Matrisi

Bunun haricinde değerlendirme metrik özetlerinde başka terimlerle de karşılaşırız.Bunlar;

Accuracy: Bu metrik bir model sonucunda en çok karşılaştığımız metriktir. Fakat sonuca doğru demek için yeterli bir metrik değildir. Bu değer modelde doğru tahmin ettiğimiz alanların tüm veri kümesine bölünmesi ile elde edilen bir değerdir.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Formül: Accueracy Formülü

Precision: Pozitif olarak tanımladığımız değerlerin kaçının pozitif olduğunu gösterir.

$$Precision = \frac{TP}{TP + FP}$$

Formül2: Precision Formülü

Recall: Bu değer bize pozitif olarak tahmin etmemiz gereken işlemlerin ne kadarını pozitif olarak tahmin ettiğimizi gösteren bir metriktir.

$$Recall = \frac{TP}{TP + FN}$$

Formül3: Recall Formülü

F1 Score: Bu değer ise, Precision ve Recall değerlerinin harmonik ortalamasını bize göstermektedir.

$$F_1 = 2\ x\ \frac{precision\ x\ recall}{precision + recall}$$

Formül4: F1 Score Formülü

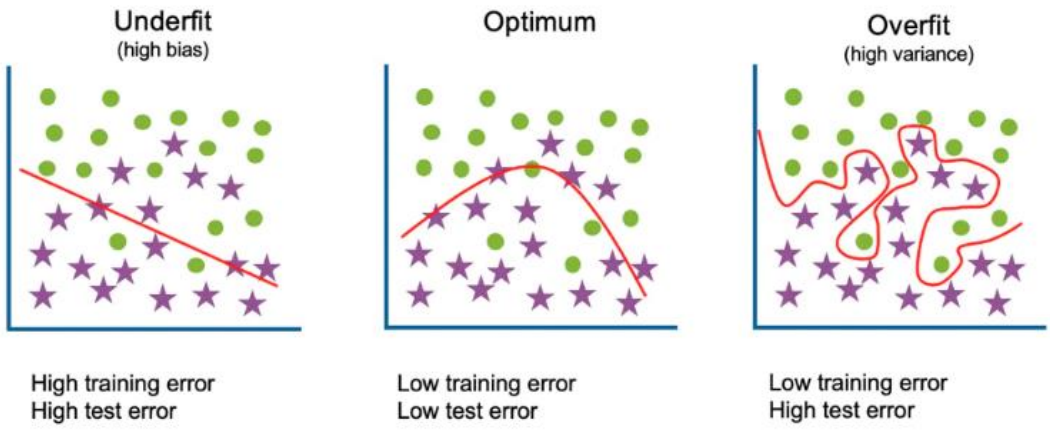
Support: Belirtilen veri kümesindeki sınıfın gerçek oluşumlarının sayısıdır. Eğitim verilerindeki dengesiz support yapısı, sınıflandırıcının rapor edilen puanlarındaki yapısal zayıflıkları gösterebilir ve tabakalı örnekleme veya yeniden dengeleme ihtiyacını gösterebilir. Support değeri, modeller arasında değişmez, bunun yerine değerlendirme sürecini teşhis eder.

```
confussion matrix
[[165649   1210]
 [  2183 164852]]


Accuracy of StackingCVClassifier: 98.98380923287031

              precision    recall  f1-score   support

           0       0.99      0.99      0.99    166859
           1       0.99      0.99      0.99    167035

    accuracy                           0.99    333894
   macro avg       0.99      0.99      0.99    333894
weighted avg       0.99      0.99      0.99    333894
```

Görüntü34: Super Learner ile Bitirilen Uygulamanın Confusion Matrisi

Overfitting

Overfitting, istatistiksel bir modelin bütünlüğü ve güvenilirliği için önemli bir tehdittir. (Cook ve Ranstam 2016). Aşırı öğrenme (over fitting), algoritmanın eğitim verisi üzerinden en alt kırılıma kadar çalışıp, sonuçları ezberlemesi ve sadece o veriler üzerinde başarı elde edebilmesidir.



Görüntü35: Underfit, Optimumve Overfit Durumları

Modelin Fazla Öğrendiğini Nasıl Anlarız?

Bunu anlamanın birçok yolu olmakla beraber bu çalışmada son uygulamadan sonra elde edilen sonuçlar için K-Katlı Çapraz Doğrulama yapılmıştır. Bu uygulama sayesinde veri seti K adet parçaya bölünür ve her parça için veri seti tekrar eğitilir. Eğer eğitilen K adet eğitim setinin test sonuçları birbirine benzer çıkıyorsa, veri setinin geneli için bu başarı sonucu ilan edilebilir. Eğer veri setinin bazı parçalarında değişken başarı durumları mevcut ise bu veri seti için bir başarı durumundan bahsedilemez ya da ilan edilen başarı tüm veri setini kapsamaz.

```
print(scores["train_score"])
```

```
[0.99050739 0.98972299 0.98988415 0.98963742 0.99007526 0.99051309
 0.99005103 0.99042896 0.99019507 0.99044893]
```

```
print(scores["test_score"])
```

```
[0.98991131 0.98924386 0.98975728 0.98934655 0.98944923 0.98973161
 0.9885506  0.99033475 0.98947477 0.98929507]
```

Görüntü36: Eğitim ve Test Seti için K-Katlı Çapraz Doğrulama

Veri Ön İşleme Adımları Olmasaydı?

```
confussion matrix
[[5918  227]
 [ 585  218]]


Accuracy of StackingCVClassifier: 88.31318364997122

              precision    recall  f1-score   support

           0       0.91      0.96      0.94      6145
           1       0.49      0.27      0.35       803

    accuracy                           0.88      6948
   macro avg       0.70      0.62      0.64      6948
weighted avg       0.86      0.88      0.87      6948
```

Görüntü37: Veri Ön İşleme Adımları Olmasaydı?

Yukarıda veri ön işleme adımlarının uygulanmadığı bir sonuç paylaşılmıştır. Burada veri ön işleme adımı olarak eksik veriler veri setinden silinmiş olup, feature selection ya da veri dengeleme gibi veri ön işleme adımları yapılmamıştır. Bu sonuçlardan da görüleceği üzere veri setinde pozitif olarak tahminleme oldukça zayıftır. Bunun sebebi ise veri setinin hedef değişkenindeki sınıf dengesizliğidir. Bu konuya önceden de değinildiği üzere dengesiz veri seti dengesiz sonuçlara yol açmıştır. Diğer bir yandan bu modelin negatif örnekleri tahminlemesi olan TN değeri de çok düşüktür. Bunun sebebi ise hem eksik verilerin veri setinde baskılanması yerine direkt çıkarılması ve aynı zamanda bu çıkarılan verilerin veri seti içerisinde dengesizliğe sebep olması olarak tanımlanabilir.

SONUÇ VE GELECEK ÇALIŞMA İÇİN ÖNERİLER

Bu çalışmada COVID19 pre-condition Data Set isimli veri setini Kaggle platformu üzerinden elde edilip, veri seti içerisinde bulunan “icu” değişkeninin hedef değişken seçilmesiyle denetimli öğrenme başlığı altında hastaların yoğun bakım ünitesine tekrar yatırılması tahmini yapılmıştır. Veri setinin ham halinde bulunan eksik değerler kategorik değişkenler için mod ile baskılanmış olup, numerik tek değişken olan “age” değişkeni için herhangi bir eksik gözlem olmadığı için bir baskılamaya ihtiyaç duyulmamıştır. Ardından ki-kare testi ile ilgili değişkenlerin hedef değişken ile bir

bağlantısı olup olmadığına bakılmıştır. Anlamsız ilişkiye sahip değişkenler veri setinden çıkarılmış olup, veri dengelenmesi adımına geçilmiştir. Bu aşamada SMOTE-N kategorik değişkenler için uygulanan bir metot olup, veri seti bu şekilde dengelenmiştir. Modelleme aşamasında veri seti GBM, XGBOOST, LightGBM modelleri tarafından eğitilmiş ve optimal parametreleri aranmıştır. Bulunan optimal parametreler ile SL (Stacking) adımına geçilip, veri seti bu uygulama ile son model haline dönüştürülmüştür. Yaklaşık %99 olan accuracy değeri overfitting şüphesiyle yaklaşmamıza fakat K-Katlı Çapraz Doğrulama ile bu şüpheden kurtulmamızı sağlamıştır. Modelin son hali için FP, FN değerleri gayet düşüktür. Precision, Recall, F1 Score ve Support değerleri de gayet dengeli ve kullanıma uygundur. Gelecek çalışmalar için, farklı bir optimal hiper parametre aramasıyla FP, FN değerleri daha da düşürülmesi yönüne gidilebilir. Kullanılan sistem özellikleri gereği daha hızlı işlem süresi ve sistemi yormaması sebebiyle seçilen RandomSearchCV yerine GridSearchCV kullanılarak, aynı zamanda tüm hiper parametreler için daha geniş belki de tanım aralığı boyunca bir değer aralığı tanımlanılıp daha kapsamlı bir arama yapılabilir. Bu sonuçları gözle görülür şekilde etkileyecektir. Başka bir çalışma kapsamı olarak, daha detaycı preprocessing adımları uygulanıp, model daha doğru sonuçlara evrilebilir.

Tüm çalışma Python üzerinden yürütülmüş olup, ilgili tüm kod blokları aşağıdaki linklerden herkes tarafından erişilebilir durumdadır.

https://github.com/erkutkoral/COVID19-pre-condition-Data-Set

https://github.com/erkutkoral/COVID19-pre-condition-Data-Set-without-Preprocessing-Steps

KAYNAKÇA

1. Keser, S. B., & Aghalarova, S. (2021). HELA: A novel hybrid ensemble learning algorithm for predicting academic performance of students. Education and Information Technologies, 1-32.
2. Mahajan, S. M., & Ghani, R. (2019, August). Using Ensemble Machine Learning Methods for Predicting Risk of Readmission for Heart Failure. In Medinfo (pp. 243-247).

3. Baillie, C. A., VanZandbergen, C., Tait, G., Hanish, A., Leas, B., French, B., ... & Umscheid, C. A. (2013). The readmission risk flag: Using the electronic health record to automatically identify patients at risk for 30-day readmission. Journal of hospital medicine, 8(12), 689-695.
4. Jamei, M., Nisnevich, A., Wetchler, E., Sudat, S., & Liu, E. (2017). Predicting all-cause risk of 30-day hospital readmission using artificial neural networks. PloS one, 12(7), e0181173.
5. Tavares, M. G., Tedesco-Silva Junior, H., & Pestana, J. O. M. (2020). Early Hospital Readmission (EHR) in kidney transplantation: a review article. Brazilian Journal of Nephrology, 42, 231-237.
6. Goto, T., Jo, T., Matsui, H., Fushimi, K., Hayashi, H., & Yasunaga, H. (2019). Machine learning-based prediction models for 30-day readmission after hospitalization for chronic obstructive pulmonary disease. COPD: Journal of Chronic Obstructive Pulmonary Disease, 16(5-6), 338-343.
7. Hemmrich, M. J., Kaskovich, S., Venable, L. R., Carey, K., Churpek, M. M., & Press, V. G. (2019). Accuracy Comparison of a Machine Learning Readmission Prediction Model with HOSPITAL and PEARL Scores for Chronic Obstructive Pulmonary Disease (COPD) Inpatients. In D102. OPTIMIZING OUTCOMES IN COPD (pp. A7118-A7118). American Thoracic Society.
8. Wallmann, R., Llorca, J., Gómez-Acebo, I., Ortega, Á. C., Roldan, F. R., & Dierssen-Sotos, T. (2013). Prediction of 30-day cardiac-related-emergency-readmissions using simple administrative hospital data. International journal of cardiology, 164(2), 193-200.
9. Dharmarajan, K., Hsieh, A. F., Lin, Z., Bueno, H., Ross, J. S., Horwitz, L. I., ... & Krumholz, H. M. (2013). Diagnoses and timing of 30-day readmissions after hospitalization for heart failure, acute myocardial infarction, or pneumonia. Jama, 309(4), 355-363.
10. Navik, U., Bhatti, J., Sheth, V., Jawalekar, S., Bhatti, G., & Kalra, S. (2020). Multi-organ failure in COVID-19 patients: a possible mechanistic approach. Authorea Preprints.

11. Hu, Y., Deng, H., Huang, L., Xia, L., & Zhou, X. (2020). Analysis of characteristics in death patients with COVID-19 pneumonia without underlying diseases. Academic Radiology, 27(5), 752.
12. Donnelly, J. P., Wang, X. Q., Iwashyna, T. J., & Prescott, H. C. (2021). Readmission and death after initial hospital discharge among patients with COVID-19 in a large multihospital system. Jama, 325(3), 304-306.
13. Chaudhry, Z., Shawe-Taylor, M., Rampling, T., Cutfield, T., Bidwell, G., Chan, X. H. S., ... & Esmail, H. (2021). Short durations of corticosteroids for hospitalised COVID-19 patients are associated with a high readmission rate. Journal of Infection, 82(6), 276-316.
14. C.0+ VV, 35, +, +3\63Ulversoy KA, Murrow JR. Impact of the COVID-19 Pandemic on Cost and Readmission Rates of Heart Failure patients at Piedmont Athens Regional.
15. Naghavi, S., Kavosh, A., Adibi, I., Shaygannejad, V., Arabi, S., Rahimi, M., ... & Ashtari, F. (2022). COVID-19 infection and hospitalization rate in Iranian multiple sclerosis patients: what we know by May 2021. Multiple Sclerosis and Related Disorders, 57, 103335.
16. Szente Fonseca SN, de Queiroz Sousa A, Wolkoff AG, Moreira MS, Pinto BC, Valente Takeda CF, et al. Risk of hospitalization for Covid-19 outpatients treated with various drug regimens in Brazil: Comparative analysis. Travel medicine and infectious disease. 2020;38:101906.
17. Shanbehzadeh, M., & Nopour, R. (2021). Determination of the most important diagnostic criteria for COVID-19: A step forward to design an intelligent clinical decision support system. Journal of Advances in Medical and Biomedical Research, 29(134), 176-182.
18. Moulaei, K., Shanbehzadeh, M., Mohammadi-Taghiabad, Z., & Kazemi-Arpanahi, H. (2022). Comparing machine learning algorithms for predicting COVID-19 mortality. BMC medical informatics and decision making, 22(1), 1-12.
19. Rojas, J. C., Carey, K. A., Edelson, D. P., Venable, L. R., Howell, M. D., & Churpek, M. M. (2018). Predicting intensive care unit readmission with machine

learning using electronic health record data. Annals of the American Thoracic Society, 15(7), 846-853.

20. Nopour, R., Shanbehzadeh, M., & Kazemi-Arpanahi, H. (2021). Developing a clinical decision support system based on the fuzzy logic and decision tree to predict colorectal cancer. Medical journal of the Islamic Republic of Iran, 35, 44.
21. Rodriguez VA, Bhave S, Chen R, Pang C, Hripcsak G, Sengupta S, et al. Development and validation of prediction models for mechanical ventilation, renal replacement therapy, and readmission in COVID-19 patients. Journal of the American Medical Informatics Association: JAMIA. 2021.
22. Huang, C. D., Goo, J., Behara, R. S., & Agarwal, A. (2020). Clinical decision support system for managing COPD-related readmission risk. Information Systems Frontiers, 22(3), 735-747.
23. Kalagara, S., Eltorai, A. E., Durand, W. M., DePasse, J. M., & Daniels, A. H. (2018). Machine learning modeling for predicting hospital readmission following lumbar laminectomy. Journal of Neurosurgery: Spine, 30(3), 344-352.
24. Chaurasia, V., & Pal, S. (2020). Application of machine learning time series analysis for prediction COVID-19 pandemic. Research on Biomedical Engineering, 1-13.
25. Ghafouri-Fard, S., Mohammad-Rahimi, H., Motie, P., Minabi, M. A., Taheri, M., & Nateghinia, S. (2021). Application of machine learning in the prediction of COVID-19 daily new cases: A scoping review. Heliyon, 7(10), e08143.
26. Shanbehzadeh, M., Orooji, A., & Kazemi-Arpanahi, H. (2021). Comparing of data mining techniques for predicting in-hospital mortality among patients with covid-19. Journal of Biostatistics and Epidemiology, 7(2), 154-173.
27. Dan, T., Li, Y., Zhu, Z., Chen, X., Quan, W., Hu, Y., ... & Cai, H. (2020, December). Machine Learning to Predict ICU Admission, ICU Mortality and Survivors' Length of Stay among COVID-19 Patients: Toward Optimal Allocation of ICU Resources. In 2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) (pp. 555-561). IEEE.
28. Lorenzen, S. S., Nielsen, M., Jimenez-Solem, E., Petersen, T. S., Perner, A., Thorsen-Meyer, H. C., ... & Sillesen, M. (2021). Using machine learning for

predicting intensive care unit resource use during the COVID-19 pandemic in Denmark. Scientific reports, 11(1), 1-10.

29. SoleimanvandiAzar, N., Irandoost, S. F., Ahmadi, S., Xosravi, T., Ranjbar, H., Mansourian, M., & Yoosefi Lebni, J. (2021). Explaining the reasons for not maintaining the health guidelines to prevent COVID-19 in high-risk jobs: a qualitative study in Iran. BMC public health, 21(1), 1-15.
30. Wirth, R., & Hipp, J. (2000, April). CRISP-DM: Towards a standard process model for data mining. In Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining (Vol. 1, pp. 29-40).
31. Belson, W. A. (1959). Matching and prediction on the principle of biological classification. Journal of the Royal Statistical Society: Series C (Applied Statistics), 8(2), 65-75.
32. Hamoud, A. (2016). Selection of best decision tree algorithm for prediction and classification of students' action. American International Journal of Research in Science, Technology, Engineering & Mathematics, 16(1), 26-32.
33. Hamoud, A., Hashim, A. S., & Awadh, W. A. (2018). Predicting student performance in higher education institutions using decision tree analysis. International Journal of Interactive Multimedia and Artificial Intelligence, 5, 26-31.
34. Nghe, N. T., Janecek, P., & Haddawy, P. (2007). A comparative analysis of techniques for predicting academic performance. Paper presented at the 2007 37th annual frontiers in education conferenceglobal engineering: knowledge without borders, opportunities without passports
35. Yadav, S. K., & Pal, S. (2012). Data mining: A prediction for performance improvement of engineering students using classification. arXiv preprint arXiv:1203.3832.
36. Asif, R., Merceron, A., Ali, S. A., & Haider, N. G. (2017). Analyzing undergraduate students' performance using educational data mining. Computers & Education, 113, 177-194.

37. Romero, C., & Ventura, S. (2010). Educational data mining: a review of the state of the art. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 40(6), 601-618.
38. Bartlett, P., Freund, Y., Lee, W. S., & Schapire, R. E. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. The annals of statistics, 26(5), 1651-1686.
39. Cao, D. S., Xu, Q. S., Liang, Y. Z., Zhang, L. X., & Li, H. D. (2010). The boosting: A new idea of building models. Chemometrics and Intelligent Laboratory Systems, 100(1), 1-11.
40. Friedman, J. H. (2002). Stochastic gradient boosting. Computational statistics & data analysis, 38(4), 367-378.
41. Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. Artificial Intelligence Review, 54(3), 1937-1967.
42. Hutagaol, N., & Suharjito, S. (2019). Predictive modelling of student dropout using ensemble classifier method in higher education. Adv. Sci. Technol. Eng. Syst. J, 4, 206-211.
43. Nagendra, K. V., Sreenivas, K., & Radhika, P. (2018). Student performance prediction using different classification algorithms.
44. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
45. Wandera, H., Marivate, V., & Sengeh, M. D. (2019). Predicting school performance using a combination of traditional and non-traditional education data from South Africa. Technical Report.
46. Calvet Liñán, L., & Juan Pérez, Á. A. (2015). Educational Data Mining and Learning Analytics: differences, similarities, and time evolution. International Journal of Educational Technology in Higher Education, 12(3), 98-112.
47. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems, 30.

48. Protikuzzaman, M., Baowaly, M. K., Devnath, M. K., & Singh, B. C. (2020). Predicting Undergraduate Admission: A Case Study in Bangabandhu Sheikh Mujibur Rahman Science and Technology University. Bangladesh, IJACSA, 11, 12.
49. Ju, C., Combs, M., Lendle, S. D., Franklin, J. M., Wyss, R., Schneeweiss, S., & van der Laan, M. J. (2019). Propensity score prediction for electronic healthcare databases using super learner and high-dimensional propensity score methods. Journal of applied statistics, 46(12), 2216-2236.
50. Kabir, M. F., & Ludwig, S. A. (2019). Enhancing the performance of classification using super learning. Data-Enabled Discovery and Applications, 3(1), 1-13.
51. Schratz, P., Muenchow, J., Iturritxa, E., Richter, J., & Brenning, A. (2019). Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. Ecological Modelling, 406, 109-120.
52. Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of machine learning research, 13(2).
53. Duarte, E., & Wainer, J. (2017). Empirical comparison of cross-validation and internal metrics for tuning SVM hyperparameters. Pattern Recognition Letters, 88, 6-11.
54. Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011, January). Sequential model-based optimization for general algorithm configuration. In International conference on learning and intelligent optimization (pp. 507-523). Springer, Berlin, Heidelberg.
55. Kuhn, M., & Johnson, K. (2013). Over-fitting and model tuning. In Applied predictive modeling (pp. 61-92). Springer, New York, NY.
56. Alghamdi, M., Al-Mallah, M., Keteyian, S., Brawner, C., Ehrman, J., & Sakr, S. (2017). Predicting diabetes mellitus using SMOTE and ensemble machine learning approach: The Henry Ford ExercIse Testing (FIT) project. PloS one, 12(7), e0179805.
57. Blagus, R., & Lusa, L. (2015). Joint use of over-and under-sampling techniques and cross-validation for the development and assessment of prediction models. BMC bioinformatics, 16(1), 1-10.

58. Chawla, N. V. (2009). Data mining for imbalanced datasets: An overview. Data mining and knowledge discovery handbook, 875-886.
59. PA, G. (2004). Batista. RC Prati, MC Monard, A study of the behaviour of several methods for balancing machine learning training data, SIGKDD Explor, 6(1), 20-29.
60. Menardi, G., & Torelli, N. (2014). Training and assessing classification rules with imbalanced data. Data mining and knowledge discovery, 28(1), 92-122.
61. Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets. International Journal of Emerging Technology and Advanced Engineering, 2(4), 42-47.
62. He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on knowledge and data engineering, 21(9), 1263-1284.
63. Poolsawad, N., Kambhampati, C., & Cleland, J. G. F. (2014, July). Balancing class for performance of classification with a clinical dataset. In proceedings of the World Congress on Engineering (Vol. 1, pp. 1-6).
64. Wang, J., Xu, M., Wang, H., & Zhang, J. (2006, November). Classification of imbalanced data by using the SMOTE algorithm and locally linear embedding. In 2006 8th international Conference on Signal Processing (Vol. 3). IEEE.
65. García, V., Alejo, R., Sánchez, J. S., Sotoca, J. M., & Mollineda, R. A. (2006, September). Combined effects of class imbalance and class overlap on instance-based classification. In International Conference on Intelligent Data Engineering and Automated Learning (pp. 371-378). Springer, Berlin, Heidelberg.
66. Jack Jr, C. R., Bernstein, M. A., Fox, N. C., Thompson, P., Alexander, G., Harvey, D., ... & Weiner, M. W. (2008). The Alzheimer's disease neuroimaging initiative (ADNI): MRI methods. Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine, 27(4), 685-691.
67. Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-validation. Encyclopedia of database systems, 5, 532-538.
68. Wayahdi, M. R., Syahputra, D., & Ginting, S. H. N. (2020). Evaluation of the K-Nearest Neighbor Model With K-Fold Cross Validation on Image Classification. INFOKUM, 9(1, Desember), 1-6.

69. Phogat, P., & Chaudhary, R. (2021, January). CURE: An effective covid-19 remedies based on machine learning prediction models. In CEUR Workshop Proceedings (Vol. 2786).
70. Benvenuto, D., Giovanetti, M., Vassallo, L., Angeletti, S., & Ciccozzi, M. (2020). Application of the ARIMA model on the COVID-2019 epidemic dataset. Data in brief, 29, 105340.
71. Jamshidi, M., Lalbakhsh, A., Talla, J., Peroutka, Z., Hadjilooei, F., Lalbakhsh, P., ... & Mohyuddin, W. (2020). Artificial intelligence and COVID-19: deep learning approaches for diagnosis and treatment. Ieee Access, 8, 109581-109595.
72. Yan, L., Zhang, H. T., Xiao, Y., Wang, M., Sun, C., Liang, J., ... & Tang, X. (2020). Prediction of survival for severe Covid-19 patients with three clinical features: development of a machine learning-based prognostic model with clinical data in Wuhan. medRxiv, 2020-02.