VERİ MADENCİLİĞİ İLE DİYABET HASTALIĞI TEŞHİSİNDE KARMA SINIFLANDIRICI ALGORİTMA ÖNERİLMESİ VE KARAR DESTEK SİSTEMİ GELİŞTİRİLMESİ

Ezgi AKTAŞ POTUR

Yüksek Lisans Tezi

Endüstri Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Nihal ERGİNEL

Eskişehir

Eskişehir Teknik Üniversitesi

Lisansüstü Eğitim Enstitüsü

Nisan 2021

ÖZET

VERİ MADENCİLİĞİ İLE DİYABET HASTALIĞI TEŞHİSİNDE KARMA SINIFLANDIRICI ALGORİTMA ÖNERİLMESİ VE KARAR DESTEK SİSTEMİ GELİŞTİRİLMESİ

Ezgi AKTAŞ POTUR

Endüstri Mühendisliği Anabilim Dalı

Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Nisan 2021

Danışman: Prof. Dr. Nihal ERGİNEL

Diyabet, uzun dönemde insan vücudunda pek çok hasara sebep olmaktadır. Dünyada 463 milyon insanın diyabet hastalığına sahip olduğu tahmin edilmektedir. Diyabetli hastaların %50,1'i hastalığından habersiz bir şekilde yaşamına devam etmektedir. Diyabet hastalarını tamamen iyileştirebilen bir tedavinin olmadığı ve hastalığından habersiz çok sayıda insanın var olduğu düşünüldüğünde erken teşhisin büyük bir öneme sahip olduğu söylenebilir. Bu çalışmada hastalık teşhisi için veri madenciliği tekniklerinden birisi olan sınıflandırma algoritmalarından faydalanılmıştır. Veri setini en iyi temsil edebilecek değişkenlerin belirlenmesi amacıyla öznitelik seçimi yapılmıştır. Seçilen öznitelikler ile sınıflandırma algoritmalarının başarıları WEKA programında test edilmiştir. En iyi sonuç veren algoritmalardan Naive Bayes, yapay sinir ağları ve karar ağacı algoritmaları birlikte kullanılarak karma bir yöntem önerilmiştir. Sınıflandırma algoritmalarının çoğunluğun tahmin sonucu dikkate alınarak oluşturulan karma yöntemin sınıflandırma başarısının algoritmaların bireysel sınıflandırma başarılarından daha yüksek olduğu görülmüştür. Visual Studio programında C# programlama dili ile veri girişi yapılmasını kolaylaştıracak görsel bir arayüz tasarlanmıştır. Oluşturulan arayüz ile kullanıcı hekimlerin öğrenmek istedikleri sonuçları karmaşıklıklar olmadan görebilmeleri sağlanmıştır.

Anahtar Sözcükler: Diyabet, Sınıflandırma algoritmaları, Hastalık teşhisi, Öznitelik seçimi, WEKA

ABSTRACT

SUGGESTION OF MIXED CLASSIFICATION ALGORITHM IN DIAGNOSIS OF DIABETES WITH DATA MINING AND DEVELOPING A DECISION SUPPORT SYSTEM

Ezgi AKTAŞ POTUR

Department of Industrial Engineering

Eskişehir Technical University, Institute of Graduate Programs, April 2021

Supervisor: Prof. Dr. Nihal ERGİNEL

Diabetes has been causing many damages in the human body in the long term. It is predicted that 463 million people in the world have diabetes. 50,1% of the patients with diabetes continue their lives unaware of their disease. Considering that there is no treatment that can completely cure diabetic patients and there are many people unaware of the disease, it can be said that early diagnosis is quite important. In this study classification algorithms which is one of the data mining techniques were used for the diagnosis of disease. Feature selection was made in order to select the variables that were best representative the data set. The success rates of classification algorithms with the selected features were tested in the WEKA. A mixed method was proposed by using Naive Bayes, artificial neural network and decision tree together, which are among the algorithms that give the best results. It was observed that the classification success of the mixed method, which was created by considering the estimation results of the majority of the classification algorithms, was higher than the individual classification success of the algorithms. A visual interface was designed in Visual Studio to facilitate data entry with C# programming language. With the interface created, the user physicians were enabled to see the results they wanted to learn without any complexity.

Keywords: Diabetes, Classification algorithms, Disease diagnosis, Feature selection, WEKA.

TEŞEKKÜR

Tez çalışması sürecinde bilgi ve tecrübeleriyle bana yol gösteren, desteğini hissettiren değerli tez danışmanım Prof. Dr. Nihal ERGİNEL'e teşekkürü borç bilirim.

Bu süreçte motivasyonumu güçlendiren, hayatımdaki tüm zorluklarda yanımda olan aileme, sevgili annem Mukaddes AKTAŞ'a, babam Hüseyin AKTAŞ'a, kardeşlerim Gizem AKTAŞ ve Gül ÖNSAL'a, tez çalışmam boyunca bilgisini ve desteğini hiçbir zaman esirgemeyen değerli eşim Caner POTUR'a sonsuz teşekkürlerimi sunarım.

Ezgi AKTAŞ POTUR

19/04/2021

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; bu çalışmanın Eskişehir Teknik Üniversitesi tarafından kullanılan "bilimsel intihal tespit programı"yla tarandığını ve hiçbir şekilde "intihal içermediğini" beyan ederim. Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

Ezgi AKTAŞ POTUR

İÇİNDEKİLER

Sayfa

TABLOLAR DİZİNİ

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

TBA	:	Temel Bileşen Analizi
AFA	:	Açıklayıcı Faktör Analizi
YSA	:	Yapay Sinir Ağı
GA	:	Güven Aralığı

1. GİRİŞ

Diyabet, pankreas adlı salgı organının yeteri kadar insülin üretemediği ya da üretilen insülinin uygun bir şekilde kullanılamadığı durumda ortaya çıkan kronik bir hastalıktır. İnsülin, yiyeceklerden gelen glikozun enerji üretmek için kan akışı yoluyla vücuttaki hücrelere geçmesini sağlayan anahtar görevli bir hormondur. İnsülin üretilemediğinde veya etkili şekilde kullanılamadığında kan şekeri yükselmektedir. Hiperglisemi olarak bilinen bu durum uzun dönemde vücutta çeşitli organ ve dokulara zarar vermektedir [1]. Damarlarda meydana gelen sertleşmeler ciddi kalp, böbrek ve göz hastalıklarını beraberinde getirmektedir.

Diyabet hastalığına bağlı ölümler dünyada ilk 4 ölüm sebebi arasındadır. Her yıl 1,6 milyon ölüm doğrudan diyabetle ilişkilidir [2]. Uluslararası Diyabet Federasyonu'nun açıklamalarına göre 463 milyon insanın diyabet hastalığına sahip olduğu tahmin edilmektedir (%95 güven aralığı: 369 milyon- 601 milyon). Diyabetli her iki insandan biri (hastaların %50,1'i) hastalığının farkında değildir. Yaklaşık yarım milyon insanın diyabetle yaşadığı göz önüne alındığında, bu hastalıkla başa çıkılabilmesi için acilen yeni stratejiler geliştirilmesine ihtiyaç duyulmaktadır. Yeterli olabilecek uygulamalar ile harekete geçilmediği takdirde 2030 yılında 578 milyon, 2045 yılında ise 700 milyon diyabetli insanın olacağı tahmin edilmektedir [3]. Diyabet hastalığının Tip I, Tip II ve Gestasyonel diyabet şeklinde farklı tipleri vardır. Tip I diyabet vücudun çok az insülin üretebildiği ya da hiç üretemediği durumları ifade etmektedir. Tüm diyabet vakalarının %90'ını oluşturan Tip II diyabet, vücut hücrelerinin üretilen insüline dirençli olduğu ve üretilen insülinin vücut hücreleri tarafından iyi kullanılamadığı durumlarda oluşmaktadır. Gestasyonel diyabet ise gebelikte oluşan yüksek kan şekeri ile ilişkilidir [1, 3].

Veri madenciliği yöntemleri, veri setlerinden örtük, daha önce bilinmeyen ve potansiyel olarak yararlı bilgilerin çıkarılması için yapılan çalışmaları içermektedir. Yararlı bilgilerin ortaya çıkarılmasında istatistiksel modeller, matematiksel algoritma ve makine öğrenme yöntemleri gibi veri analiz araçları kullanılmaktadır. Makine öğrenmesi ile çeşitli amaçlar için kullanılacak bilgiler veri tabanlarındaki ham verilerden anlaşılır biçimde elde edilebilmektedir. Sınıflandırma, verilerin daha önceden belirlenmiş sınıflardan hangisine ait olduğunun tespit edilmesinde kullanılan bir makine öğrenmesi tekniğidir [4, 5]. Günümüzde pek çok alanda uygulamaları bulunan veri madenciliği yöntemlerinin en çok kullanıldığı alanlardan birisi de tıptır. Sınıflandırma algoritmaları

tıpta bazı hastalıkların teşhis edilmesinde kullanılabilmektedir. Günümüzde veri madenciliği hayati bir role sahiptir ve tıbbi teşhis için temel bir metodoloji haline gelmiştir. Geçmişten günümüze sınıflandırma yöntemleri kullanılarak diyabet hastalığının tespit edilmesine yönelik çeşitli çalışmalar yapılmıştır. Bununla birlikte diyabetin doğru bir şekilde tespit edilebilmesi için yeni yöntemlere ihtiyaç duyulmaktadır [6, 7]. Geçmişte yapılan çalışmaların çoğunluğu en iyi sınıflandırma yönteminin belirlenmesine yöneliktir.

Bu tez çalışmasında Tip II diyabet hastalarının hastalıklarının tespit edilebilmesi için birden fazla sınıflandırma algoritması birlikte kullanılmış ve öznitelik seçimi yapılarak öznitelik seçiminin sınıflandırma başarısı üzerindeki etkisi incelenmiştir. Öznitelik seçim yöntemi olarak kullanılan PROC VARCLUS yöntemi ile değişkenler kümelere ayrılmıştır. Oluşturulan kümelerden varyasyonu en çok açıklayan değişkenler temsilci olarak seçilmiştir. Uygulanan sınıflandırma algoritmalarından çeşitli değerlendirme ölçütlerine göre en iyi sonuç veren üç yöntem belirlenmiştir ve bu yöntemlerin birlikte değerlendirilmesiyle oluşturulan karma yöntemin sonuçlarının hesaplanmasında çoğunluğun oyu dikkate alınmıştır. Çalışmada daha az sayıda öznitelik kullanılması ve sınıflandırma algoritmalarının birlikte değerlendirilmesi ile sınıflandırma başarısının arttırılması amaçlanmıştır. Öznitelik seçiminin kullanılan sınıflandırma algoritmalarının performansları üzerindeki etkileri incelenmiş ve çalışma kapsamında karma yöntemden elde edilen sonuçların kullanılan algoritmalardan ayrı ayrı elde edilen sonuçlar ile karşılaştırılması yapılmıştır.

Çalışmanın izleyen bölümünde literatür incelemesine yer verilmiştir. Üçüncü bölümünde veri madenciliği tekniklerinden sınıflandırma algoritmalarına yer verilmiştir ve öznitelik seçim yöntemleri hakkında bilgi verilmiştir. Dördüncü bölümde diyabet hastalığı, tipleri ve ortaya çıkabilecek komplikasyonlardan bahsedilmiştir. Beşinci bölümünde diyabet hastalığının teşhisi için yeni bir veri seti ile sınıflandırma algoritmaları birlikte kullanılarak geliştirilen karma yönteme ve karar destek sistemine yer verilmiştir. Altıncı bölümünde çalışmadan elde edilen sonuçlar aktarılmış, bu sonuçlar ile ilgili değerlendirmeler yapılmış ve gelecek çalışmalar için önerilerde bulunulmuştur.

2. LİTERATÜR İNCELEMESİ

Son zamanlarda kullanımı giderek yaygınlaşan yöntemlerden birisi olan veri madenciliği ile ilgili birçok alanda önemli çalışmalar yapılmıştır ve yeni yöntemlerin geliştirilmesiyle birlikte farklı çalışmalar yapılmaya devam etmektedir. Veri madenciliği tekniklerinden sınıflandırma algoritmalarında, daha verimli sonuçlar elde edilebilmesi için öznitelik seçimi yöntemlerinden yararlanılmaktadır. Bu kapsamda uygulanabilecek yöntemlerden birisi kümeleme tabanlı öznitelik seçimidir. Literatürde çeşitli hastalıkların ve diyabet hastalığının teşhis edilmesi konusunda öznitelik seçimi yöntemlerinin ve sınıflandırma algoritmalarının kullanıldığı çok sayıda bilimsel çalışma bulunmaktadır.

Ismi, Panchoo ve Murinto (2016) [8] tarafından yapılan çalışmada UCI web sitesinde bulunan 7352 örnek ve 561 öznitelik içeren akıllı telefon ile aktivite tanıma veri seti ile 1080 örnek ve 857 öznitelik içeren ulusal ekonomik aktivitelerin sınıflandırılması veri setleri kullanılmıştır. Sınıflandırma işlemi öncesinde benzer özniteliklerin bir arada olması amacıyla k-ortalamalar yöntemi ile öznitelikler kümelenmiştir. Yüksek hesaplama zamanlarının ve iş yükünün azaltılması amacıyla aynı kümede bulunan öznitelikler, küme merkez değerleri ile temsil edilmiştir. Farklı k değerleri seçilerek Naive Bayes sınıflandırma algoritması ile sınıflandırma yapılmıştır. Akıllı telefon ile aktivite tanıma veri setinde seçilen k değerleri içerisinde en yüksek doğru sınıflandırma oranı 70 öznitelik kullanıldığı durumda %83.35 olarak elde edilmiştir. Diğer veri seti için en yüksek doğru sınıflandırma oranı 60 öznitelik için %88.33 olarak hesaplanmıştır.

Patrício vd. (2018) [9] tarafından yapılan çalışmada 116 kişiye ait yaş, vücut kitle indeksi ve çeşitli klinik özellikler incelenerek hastalarda göğüs kanserinin varlığı için bir tahmin modeli oluşturulması, tanı kolaylığının arttırılması amaçlanmıştır. Çalışmada destek vektör makineleri, lojistik regresyon ve rastgele orman algoritmaları kullanılmıştır. Resistin, glikoz, yaş ve vücut kitle indeksi değişkenleri ile yapılan sınıflandırmada kullanılan değerlendirme ölçütleri için en başarılı sonuçlar destek vektör makineleri ile yapılan sınıflandırmada elde edilmiştir.

Sobar ve Wijaya (2016) [10] tarafından yapılan çalışmada rahim ağzı kanserinin erken teşhis edilebilmesi için 22'si hastalığa sahip olan ve 50'si sağlıklı olan 72 kişinin bilgilerini içeren bir veri seti kullanılmıştır. Katılımcılara anket yöneltilmiş ve verdikleri cevaplar kaydedilmiştir. 19 adet öznitelik kullanılarak yapılan sınıflandırmada Naive Bayes ve lojistik regresyon sınıflandırma yöntemleri kullanılmıştır. Yöntemler

karşılaştırıldığında Naive Bayes yöntemi ile daha başarılı bir sınıflandırma yapıldığı görülmüştür.

Mumtaz vd. (2017) [11] tarafından major depresif bozukluğun teşhisi için yapılan çalışmada 34'ü hasta, 30'u sağlıklı olmak üzere 64 kişinin EEG kayıtları kullanılarak hastalığın teşhis edilmesi amaçlanmıştır. Çalışmada kullanılan EEG kayıtları hastaların 5 dakika gözlerinin kapalı, 5 dakika gözlerinin açık olduğu durumları içermektedir. Beyinde gerçekleşen elektriksel aktiviteler kafa derisine yerleştirilen 19 elektrot ile ölçülmüştür. Farklı frekanslardaki beyin dalgaları kullanılarak yapılan sınıflandırmalarda lojistik regresyon, Naive Bayes ve destek vektör makineleri yöntemlerinin sınıflandırma başarıları değerlendirilmiştir. Değerlendirme ölçütleri için algoritmalar arasında bir karşılaştırma yapıldığında destek vektör makinelerinin en yüksek sınıflandırma başarısına sahip olduğu görülmüştür.

Azam vd. (2020) [12] tarafından yapılan çalışmada UCI web sitesinden alınan 303 örneği içeren Cleveland kalp hastalığı veri seti kullanılmıştır. Çalışmada Naive Bayes, karar ağacı, lojistik regresyon ve rastgele orman sınıflandırıcıları kullanılmıştır. Daha az sayıda bir grup değişken kullanılarak tahmin yapılabilmesi için boyut azaltma tekniklerinden temel bileşenler analizinden yararlanılmıştır. 2 adet temel bileşen oluşturulmuştur. Çalışma kapsamındaki 14 öznitelik ile yapılan sınıflandırmadan elde edilen sonuçlar temel bileşenler kullanılarak elde edilen sonuçlar ile karşılaştırılmıştır. Değerlendirme ölçütleri tüm sonuçlar için dikkate alındığında en iyi kombinasyonun temel bileşenler kullanılmadan rastgele orman algoritması ile yapılan sınıflandırmada elde edildiği görülmüştür.

Das, Naik ve Behera (2018) [6] tarafından yapılan çalışmada diyabet hastalığının erken teşhisi için sınıflandırma yöntemlerinden J48 karar ağacı yöntemi ve Naive Bayes yöntemi karşılaştırılmıştır. Veri toplama aracı olarak cinsiyet, kan şekeri seviyesi, plazma glikoz konsantrasyonu gibi bilgileri içeren 7 soruluk bir form hazırlanmıştır. Her sorunun bir özniteliği temsil ettiği formda kullanılan sorular 200 kişiye yöneltilmiştir. Naive Bayes yönteminin daha yüksek doğru sınıflandırma oranına sahip olduğu ve J48 yöntemine göre daha verimli bir performans gösterdiği görülmüştür.

Alassaf vd. (2018) [13] tarafından yapılan çalışmada yapay sinir ağı, Naive Bayes, destek vektör makineleri, k-en yakın komşu algoritması olmak üzere dört farklı

sınıflandırıcı değerlendirilmiştir. 399 örnek ve glikoz, yaş, MPV, HCT, HGB, RBC, EOS, MCH, kreatinin, keton gibi 48 özniteliği içeren veri seti King Fahd Üniversite Hastanesinden alınmıştır. En iyi doğru sınıflandırma oranı HCT, MPV ve glikoz özellikleri kullanılarak elde edilmiş ve doğru sınıflandırma oranı, kesinlik, duyarlılık ve F ölçütü değerlendirme kriterleri için en yüksek sınıflandırma başarısına sahip yöntem yapay sinir ağı olmuştur.

Nimmala (2019) [14] tarafından yapılan çalışmada 50 kişiye ait gerçek hayat verileri kullanılarak yaş, obezite ve kolesterol seviyesinin diyabet üzerindeki etkileri analiz edilmiştir. Doğru sınıflandırma oranı, ortalama mutlak hata, göreceli mutlak hata ve kök göreceli kare hata değerlendirme kriterleri JRip, Naive Bayes, J48 karar ağacı ve rastgele orman sınıflandırıcıları için değerlendirilmiştir. Araştırma sonuçlarına göre diyabetli hastalarda obezite ve kolesterol düzeyleri oldukça farklılık göstermektedir. En düşük göreceli mutlak hata ve kök göreceli hata değerleri rastgele orman sınıflandırıcısı ile elde edilmiştir. En düşük ortalama mutlak hataya ve en yüksek doğru sınıflandırma oranına sahip Naive Bayes algoritması ile % 76 doğru sınıflandırma oranına ulaşılmıştır.

Daghistani ve Alshammari (2020) [15] tarafından yapılan çalışmada demografik özellikler, yaş, vücut kitle indeksi, kan basıncı ve MCH, MCV, MCHC, RBC, HGB, HCT gibi laboratuvar testleri içeren 18 öznitelik kullanılarak diyabet hastalığının tahmin edilmesi için karar ağacı ve lojistik regresyon algoritmalarının performansları karşılaştırmışlardır. Karar ağacı algoritması ile doğru sınıflandırma oranı %88, F ölçütü 0.876, kesinlik 0.883 ve duyarlılık 0.88 olarak hesaplanmıştır. Tüm değerlendirme ölçütleri için karar ağacı algoritması ile lojistik regresyon algoritmasına göre daha iyi sonuçlar elde edilmiştir.

Selvakumar vd. (2017) [16] tarafından yapılan çalışmada 7 farkı değişken içeren 100 kişinin verilerinin bulunduğu bir diyabet veri seti oluşturulmuştur. Diyabet hastalığının tahmin edilmesi için çok katmanlı algılayıcı (yapay sinir ağı), ikili lojistik regresyon ve k en yakın komşu sınıflandırıcılarının doğru sınıflandırma oranları karşılaştırılmıştır. En yüksek doğru sınıflandırma oranını %80 ile k en yakın komşu yöntemi verirken izleyen sınıflandırıcılar %71 ile çok katmanlı algılayıcı ve %69 ile ikili lojistik regresyon olmuştur.

Diyabet hastalığının teşhisi için yapılan çoğu çalışmada Pima yerlileri diyabet veri setinin kullanıldığı görülmüştür.

Iyer, Jeyalatha ve Sumbaly'nin (2015) [17] çalışmasında Pima yerlileri diyabet veri seti kullanılarak yapılan çalışmada J48 karar ağacı ve Naive Bayes algoritmalarının performanslarını karşılaştırılmıştır. WEKA yazılımında bulunan filtre tabanlı CfsSubsetEval (korelasyon tabanlı öznitelik seçimi) yöntemi kullanılmıştır. Öznitelikler, öznitelik alt kümeleri kullanılarak korelasyona dayalı sezgisel bir değerlendirme fonksiyonuna göre sıralanmış ve uygun öznitelikler seçilmiştir. Naive Bayes algoritması için veri seti %70 eğitim /%30 test verisi olarak ayrılarak hesaplanırken karar ağacı algoritması sonuçlar %70 eğitim /%30 test ve 10 kat çapraz doğrulama olarak iki farklı şekilde hesaplanmıştır. Hesaplanan sonuçlar karşılaştırıldığında Naive Bayes yöntemini en düşük hata oranına sahip olduğu görülmüştür.

Kurt ve Ensari (2017) [18] tarafından yapılan çalışmada UCI web sitesindeki Pima yerlileri veri seti kullanılarak diyabet hastalığının teşhisi için yapay sinir ağı, lineer, polinomsal ve radyal tabanlı destek vektör makineleri sınıflandırma yöntemlerinin başarıları değerlendirilmiştir. Lineer destek vektör makineleri %77.47 başarı oranı ile doğru teşhis oranı en yüksek olan yöntem olmuştur.

Kumari ve Chitra (2013) [19] tarafından Pima yerlileri diyabet veri seti kullanılarak yapılan çalışmada destek vektör makinelerinin diyabet hastalığının teşhisindeki başarısı değerlendirilmiştir. Çalışmada 460 kişiye ait veri ve numerik yapıda 8 öznitelik kullanılmıştır. Radyal tabanlı çekirdek fonksiyonun kullanıldığı çalışmada destek vektör makinelerinin doğru sınıflandırma oranı, duyarlılık ve özgüllük parametreleri sırasıyla %78.20, %80 ve %76.50 olarak bulunmuştur.

Kaur ve Chhabra (2014) [20] tarafından yapılan çalışmada Pima yerlileri diyabet veri seti kullanılarak J48 Karar ağacı algoritması üzerinde modifiye yapılmıştır. Uygulama programlama arayüzü olarak WEKA kullanılmıştır. Verilerin bulunduğu Arff formatındaki dosya MATLAB'a aktarılmıştır. Çok sayıda özelliğe erişim sağlanması ile MATLAB programlama dilinde düzenlenen modifiye yöntemden elde edilen deneysel sonuçların %99.870 doğru sınıflandırma oranı ile %73.828 doğru sınıflandırma oranına sahip mevcut J48 algoritmasına göre önemli ölçüde gelişme gösterdiği ve sırasıyla %76.302, %73.391, %68.099 doğru sınıflandırma oranına sahip Naive Bayes, çok

katmanlı algılayıcı, rastgele ağaç gibi sınıflandırıcılara göre daha yüksek bir başarı oranını yakaladığı görülmüştür.

Bozkurt vd. (2014) [21] tarafından yapılan çalışmada Pima yerlileri diyabet veri seti 6 farklı yapay sinir ağı, yapay bağışıklık sistemi ve Gini algoritması kullanılarak oluşturulan karar ağacı sınıflandırıcılarının başarılarının hesaplanması için kullanılmıştır. Değerlendirmeye alınan toplamda 8 sınıflandırıcıdan en yüksek doğruluk ve özgüllük değerlerine sahip sınıflandırıcı %76.00 doğruluk ve %88.75 özgüllük değeri ile dağıtılmış zaman gecikmeli ağlar olurken en yüksek duyarlılık değerine sahip sınıflandırıcı %63.33 duyarlılık değeri ile olasılıklı yapay sinir ağı olmuştur.

Garg vd. (2018) [22] tarafından Pima yerlileri diyabet veri seti kullanılarak yapılan araştırmada karar tabloları, Naive Bayes, Bayes ağları, J48 karar ağacı, destek vektör makineleri ve rastgele orman karar ağacı sınıflandırıcılarının sınıflandırma başarıları karşılaştırılmıştır. 10-kat çapraz doğrulama yapıldığında destek vektör makineleri %77.34 doğru sınıflandırma oranı ve %22.65 hatalı sınıflandırma oranı ile iyi sınıflandırıcı olurken eğitim ve test verisi bölündüğünde en yüksek doğru sınıflandırma oranı %81.99 ile karar tabloları olmuştur.

Radha ve Srinivasan (2014) [23] tarafından yapılan çalışmada UCI makine öğrenmesi veri tabanından elde edilen Pima yerlileri diyabet veri seti ile Hindistan Tıbbi Araştırma Konseyi'nden elde edilen veriler kullanılmıştır. C4.5, destek vektör makineleri, k en yakın komşu, yapay sinir ağı ve ikili lojistik regresyon sınıflandırıcıları kullanılmıştır. En yüksek doğru sınıflandırma oranına sahip yöntem %86 ile C4.5 karar ağacı olmuştur. Hesaplama süresi dikkate alındığında %75 doğru sınıflandırma oranına sahip ikili lojistik regresyon yönteminin en düşük hesaplama süresine sahip olduğu görülmüştür.

Akyol ve Şen (2018) [24] tarafından Pima yerlileri veri seti kullanılarak diyabetli ve sağlıklı hastaların ayrımının yapılması için etkili özniteliklerin seçimi ve sınıflandırma olmak üzere iki aşamadan oluşan bir çalışma yapılmıştır. Öznitelik seçimi için Recursive Feature Elimination (RFE), Stability Selection (SS) ve Iterative Relief (IR) yöntemleri uygulanmıştır. Sınıflandırma için Adaboost, Gradient Boosted ağaçlar ve rastgele orman algoritmaları kullanılmıştır. Stability Selection (SS) öznitelik seçim yöntemi kullanılarak

uygulanan Adaboost öğrenme algoritmasının %73,88 doğru sınıflandırma oranına sahip olduğu ve diğer iki yönteme göre daha başarılı sonuç verdiği görülmüştür.

Kumar ve Umatejaswi'nin (2017) [25] diyabet hastalığının teşhisi ile ilgili yaptığı çalışmada 650 kişiye ait veri seti k-ortalamalar kümeleme yöntemiyle gestasyonel diyabet, tip-1 ve tip-2 diyabet olarak 3 kümeye ayrılmıştır. Kümelenen veri seti oluşturulan sınıflandırma modellerine girdi olarak verilerek hastaların risk seviyelerinin hafif orta ve şiddetli olarak sınıflandırılmasında kullanılmıştır. Veri ön işleme kısmında veri setinin tutarsız verilerden arındırılması için filtreleme yapılmış ve bunun sonucunda 620 hastaya ait veri girdi olarak kullanılmıştır. Sınıflandırma işleminde rastgele ağaç, Naive Bayes, C4.5 karar ağacı ve doğrusal lojistik regresyon sınıflandırma algoritmaları kullanılmıştır. Algoritmaların hata oranlarına ve doğru sınıflandırma oranlarına bakıldığında en başarılı sonuç %0 hata oranı ve %100 doğru sınıflandırma oranı ile C4.5 karar ağacı algoritmasından alınmıştır.

Bashir vd. (2014) [26] tarafından yapılan çalışmada BioStat ve UCI web sitelerindeki diyabet veri setleri kullanılmıştır. ID3, C4.5, CART olmak üzere üç farklı karar ağacı temel alınarak majority voting, adaboost, Bayesian Boosting, stacking ve bagging teknikleri uygulanmıştır. Doğru sınıflandırma oranı, duyarlılık, özgüllük ve f ölçütü değerlendirme ölçütlerine göre bagging yöntemi sırasıyla %91.56, %95.63, %68.33, %79.71 sonuçlarını vermiştir. Bagging yöntemi ile yapılan sınıflandırmada doğru sınıflandırma oranı diğer yöntemlere göre daha yüksek bulunmuştur. Fakat yöntemler arasında duyarlılık, özgüllük ve f ölçütü değerlerinin birbirine oldukça yakın olduğu görülmüştür.

Zhu vd. (2015) [27] tarafından çoklu sınıflandırıcı kullanılmasına dayalı yapılan çalışmada Pima yerlileri ve Endonezya veri seti kullanılarak Tip 2 diyabet hastalığının teşhisi için ağırlık oylama sistemine dayanan bir mantık geliştirilmiştir. Değerlendirmede doğru sınıflandırılan örneklerin yanı sıra ROC eğrileri de incelenmiştir. Çoklu sınıflandırıcı oluşturulurken destek vektör makineleri, C4.5 karar ağacı, Naive Bayes, lojistik regresyon ve sinir ağları kullanılmıştır. Deneysel sonuçlar önerilen yöntemin Tip 2 diyabet hastalığının tespiti açısından daha iyi performans gösterdiğini ortaya çıkarmıştır.

Dewangan ve Agrawal (2015) [28] tarafından Pima yerlileri diyabet veri seti kullanılarak yapılan çalışmada C4.5 karar ağacı, bayes ağları, rastgele orman ve çok katmanlı algılayıcı sınıflandırıcıları kullanılmıştır. Veri setindeki 8 farklı özniteliğin önem derecesine göre sıralanması ile gereksiz öznitelikler ortadan kaldırılmıştır. Farklı eğitim-test yüzdelerinin sınıflandırma başarısını etkilediği görülmüştür. Örneğin C4.5 algoritması için eğitim test verisi %75-%25 olarak ayrıldığında %77.08 doğru sınıflandırma oranını verirken %85-%15 olarak ayrıldığında %76.72 ve %90-%10 olarak ayrıldığında %75.32 doğru sınıflandırma oranını vermiştir. C4.5 ve rastgele orman algoritmaları birlikte kullanıldığında doğru sınıflandırma oranı %79.31 olurken çok katmanlı algılayıcı ve bayes ağları birlikte kullanıldığında %81.89 doğru sınıflandırma oranına ulaşılmıştır. İkili kombinasyonların bireysel sonuçlardan daha yüksek başarı oranlarına sahip olduğu ve en başarılı kombinasyonun %81.89 doğru sınıflandırma oranı, %64.10 duyarlılık ve %90.9 özgüllük değerleri ile çok katmanlı algılayıcı ve bayes ağları kombinasyonu olduğu görülmüştür.

Devi vd. (2020) [29] tarafından yapılan çalışmada Pima yerlileri veri setindeki veriler diyabetli ve diyabetli olmayan olarak sınıflandırılmıştır. Verilerin gruplanması için ağırlık merkezi seçimi ve küme ataması aşamalarından oluşan en uzak ilk kümeleme algoritması (farthests first clusterig) ve gruplanan verilerin sınıflandırılması için kullanılan sıralı minimum optimizasyon algoritmasının entegre edilmesi ile diyabet hastalığının teşhisi için bütünleşik bir yaklaşım önerilmiştir. Bütünleşik yaklaşımın sonuçlarına göre doğru sınıflandırma oranı %99.4, bağıl karesel hata karekökü (RRSE) 100.156, hata kareleri ortalamasının karekökü (RMSE) 0.074, ortalama mutlak hata (MAE) 0.005 ve bağıl mutlak hata (RAE) 44.209 olarak elde edilmiştir. Gruplama sonucunda hesaplama süresinin kısaldığı ve sınıflandırma işleminin yüksek doğrulukla gerçekleştirildiği görülmüştür.

Kannadasan vd. (2019) [7] tarafından Pima yerlileri veri setindeki verilerin sınıflandırılması için derin sinir ağı sınıflandırma yöntemi uygulanmıştır. Çalışmada yığılı oto kodlayıcılar kullanılarak öznitelikler çıkarılmış ve softmax katman kullanılarak veri kümesi sınıflandırılmıştır. Tip 2 Diyabet verilerinin sınıflandırılması için yığılı oto-kodlayıcı tabanlı derin öğrenme yaklaşımına göre veriler %86.26 doğrulukla sınıflandırılmıştır. Kesinlik, duyarlılık, özgüllük ve f ölçütü metrikleri sırasıyla 90.66, 87.92, 83.41 ve 89.27 olarak hesaplanmıştır.

Veri madenciliği teknikleri kullanılarak diyabet tahminine ek olarak, hemogram parametreleri ve kan lipitlerinin diyabetik hastalar üzerindeki etkilerinin kontrol gruplarıyla karşılaştırılarak analiz edildiği farklı çalışmalar yapılmıştır. Bu kapsamdaki çalışmalar incelendiğinde HGB, RBC, MCH, MCV, MCHC, PDW parametreleri açısından diyabetik grup ile kontrol grubu arasında istatistiksel olarak anlamlı fark olduğu görülmüştür ($p < 0.05$) [30, 31] .

Literatürdeki diyabet hastalığının teşhisi için yapılan çalışmaların çoğunda Pima yerlileri veri setinin kullanıldığı ve çalışmaların çoğunluğunun en yüksek doğru sınıflandırma oranını veren yöntemin seçimi ile ilgili olduğu görülmüştür. Pima yerlileri veri seti hamilelik ile ilgili bilgiler içerdiğinden ve hamilelikte kan değerlerinde değişiklikler görülebileceğinden diyabet hastalığının teşhis edilmesi için yeni ve gerçek bir veri seti ile çalışılmıştır. PROC VARCLUS yöntemi ile sınıflandırma üzerinde en çok etkili olan özniteliklerin seçimi yapılmıştır. Veri seti üzerinde sınıflandırma başarısı yüksek olan birden fazla algoritmanın oylama yöntemi ile birlikte değerlendirilmesi sonucunda bireysel yöntemlerden daha yüksek sınıflandırma başarısına sahip karma bir yöntem oluşturulmuştur. Ayrıca diyabet hastalığının teşhisi için yapılan çoğu çalışmadan farklı olarak oluşturulan karma yöntem ile gerçekleştirilen sınıflandırmaya ait duyarlılık, kesinlik ve doğru sınıflandırma oranı değerlendirme ölçütleri için güven aralıkları hesaplanmıştır.

3. VERİ MADENCİLİĞİ

Günümüzde pek çok kaynak tarafından her gün büyük miktarlarda veri üretilmekte ve depolanmaktadır. Veri kaynakları veri tabanlarını, veri ambarlarını, bilgi havuzlarını veya sisteme dinamik olarak aktarılan verileri içerebilmektedir. Üretilen verilerin analiz edilmesi ve değerli bilgilerin keşfedilmesi için güçlü ve çok yönlü araçların gerekliliği veri madenciliğinin doğuşunda etkili olmuştur. Veri madenciliği, 1980'li yıllardan itibaren güçlü bilgisayarlar, veri toplama ekipmanları ve artan depolama ortamı kaynakları ile hızlı bir şekilde ilerleme göstermiştir. Veri madenciliği faydalı sonuçlar elde edilebilmesi amacıyla başlangıçta bilinmeyen ilişkilerin ve örüntülerin keşfedilebilmesi için büyük miktarda verinin seçilmesi, araştırılması ve modellenmesi sürecidir [32, 33].

3.1. Veri Madenciliği Süreci

Veri madenciliği süreci izleyen adımlardan oluşmaktadır [33]:

1.Veri temizleme: Gürültü ve tutarsız verilerin giderildiği aşamadır.

2. Veri entegrasyonu: Birden fazla veri kaynağının birleştirilebildiği durumlarda uygulanan aşamadır.

3. Veri seçimi: Yapılacak analiz ile ilgili verilerin veri tabanından seçiminin yapıldığı aşamadır.

4. Veri azaltma ve dönüştürme: Boyut azaltma için özelliklerin (özniteliklerin) seçildiği, verilerin özet veya birleştirme işlemleri ile veri madenciliği için uygun formlara dönüştürüldüğü aşamadır.

5. Veri madenciliği: Veri örüntülerinin ortaya çıkarılabilmesi için modelleme ve akıllı metotların uygulandığı temel bir süreçtir.

6. Model Değerlendirme: Bilgiyi temsil eden modelin değerlendirilmesi ve örüntülerin tanımlanmasıdır.

7. Bilgi Sunumu: İşlenen verinin görselleştirme ve bilgi sunum teknikleri ile kullanıcılara sunulduğu aşamadır.

1'den 4'e kadar olan adımlar, verilerin hazırlandığı farklı veri ön işleme biçimleridir.

3.2. Sınıflandırma Algoritmaları

Sınıflandırma, sınıf etiketi bilinmeyen veri nesnelerinin sınıflarının tahmin edilebilmesi için sınıf etiketi bilinen nesnelerin analizine dayalı, kavramları tanımlayan ve ayıran bir model bulma sürecidir. Verilerin ait oldukları sınıfı gösteren değerlere etiket denmekte, eğitim verisinin analizi ile oluşturulan modelin başarımı test verisi ile belirlenmektedir. Model başarımı, doğru sınıflandırılmış test kümesi örneklerinin oranı ile hesaplanmaktadır [33].

3.2.1. Naive Bayes sınıflandırma algoritması

Temeli Thomas Bayes'in Bayes teoremine dayanan Naive Bayes algoritması, sonuç değişkeninin kategorilere göre sınıflara ayrıldığı durumlarda koşullu sınıf olasılıklarının tahmin edilerek ilgilenilen bir durumun hangi sınıfa ait olduğunun belirlenmesinde kullanılmaktadır. Naive Bayes algoritmasında öznitelikler birbirinden bağımsız kabul edilmektedir. Koşullu bağımsızlık varsayımı şu şekilde ifade edilmektedir [34]:

$$\mathrm{P(X|Y=y)} = \prod_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{P(X_i|Y=y)} \tag{3.1}$$

Burada Y sınıf etiketini ve X n adet özniteliğin oluşturduğu kümeyi (X=$\{X_1, X_2, \dots, X_n\}$) göstermektedir. Naive Bayes sınıflandırıcısı ile verilerinin sınıf etiketlerinin tahmin edilmesinde her sınıf için hesaplanan koşullu olasılık değeri (3.2) denklemi ile ifade edilmiştir.

$$(Y|X) = \frac{P(Y)\prod_{i=1}^{n} P(X_i|Y)}{P(X)} \tag{3.2}$$

Her Y için P(X) sabit olduğundan (3.2) denkleminde pay terimini en büyükleyen sınıfın seçilmesi yeterlidir. X_i özniteliğinin aldığı kategorik bir değer için hesaplanan koşullu olasılık $P(X_i = x_i|Y = y)$ y sınıfı içinde belirli bir x_i öznitelik değerini alan eğitim örneklerinin oranı ile hesaplanmaktadır. X_i özniteliğinin aldığı sürekli bir değer için belirli bir olasılık dağılımı varsayılabilmektedir. Eğitim verileri kullanılarak dağılımın parametreleri tahmin edilmektedir. Sürekli özniteliklerin koşullu sınıf olasılıklarının tahmininde çoğunlukla Gauss (normal) dağılımı kullanılmaktadır. Dağılımın aritmetik ortalama ve varyans olmak üzere iki parametresi vardır. Her y_j sınıfı için X_i özniteliğinin koşullu sınıf olasılığı denklem (3.3)'te verilmiştir.

$$\left(X_i = x_i \middle| Y = y_j\right) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(x_j-\mu_{ij})^2}{2\sigma_{ij}^2}} \quad (3.3)$$

3.2.2. K-en yakın komşu algoritması

K-en yakın komşu algoritması yeni bir test verisinin eğitim setindeki örneklerle karşılaştırılması ve aralarındaki benzerliklere göre bir sınıfa atanmasına dayanan örnek tabanlı bir öğrenme algoritmasıdır. Veri seti n adet öznitelik ile tanımlandığında her bir örnek, n boyutlu uzayda bir noktayı temsil etmektedir. Sınıf etiketi bilinmeyen yeni bir örnek geldiğinde yeni örneğe en yakın k adet komşunun belirlemesi için n boyutlu eğitim örnek uzayı taranmaktadır. Yeni örnek, belirlenen k en yakın komşusunun çoğunluğunun ait olduğu sınıfa atanmaktadır [33]. K en yakın komşu algoritmasında komşu sayısını gösteren k'nın aldığı değere bağlı olarak örneğin sınıfı belirlenmektedir. Sınıflandırma yapılırken k=1 olarak belirlendiğinde yeni gelen örnek en yakın komşusunun bulunduğu sınıfa atanmaktadır. Yakınlık ifadesi Öklid mesafesi gibi bir mesafe cinsinden tanımlanmaktadır. Öklid uzaklığı sınıflandırma algoritmalarında sıklıkla kullanılan uzaklıklardandır. Bunun dışında Minkowski, Manhattan, Chebyschev gibi farklı uzaklık ölçüleri de kullanılabilmektedir. A= (x_1, x_2,…,x_n) ve B= (y_1, y_2,…,y_n) olmak üzere A ve B noktaları arasındaki Minkowski uzaklığı şu şekilde ifade edilmektedir [35]:

$$Minkowski = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}} \quad (3.4)$$

(3.4) numaralı denklemde p'nin aldığı değerlere göre farklı uzaklık ölçütleri elde edilebilmektedir. Uzaklık hesaplamasında p=2 olduğunda Öklid uzaklığı, p=1 olduğunda Manhattan uzaklığı ve n→ ∞ oluğunda ise Chebyschev uzaklığı elde edilmektedir. A ve B noktaları arasındaki Öklid, Minkowski ve Chebyschev uzaklıkları şu şekilde ifade edilmektedir [35]:

$$Öklid = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \quad (3.5)$$

$$Manhattan = \sum_{i=1}^{n} |x_i - y_i| \quad (3.6)$$

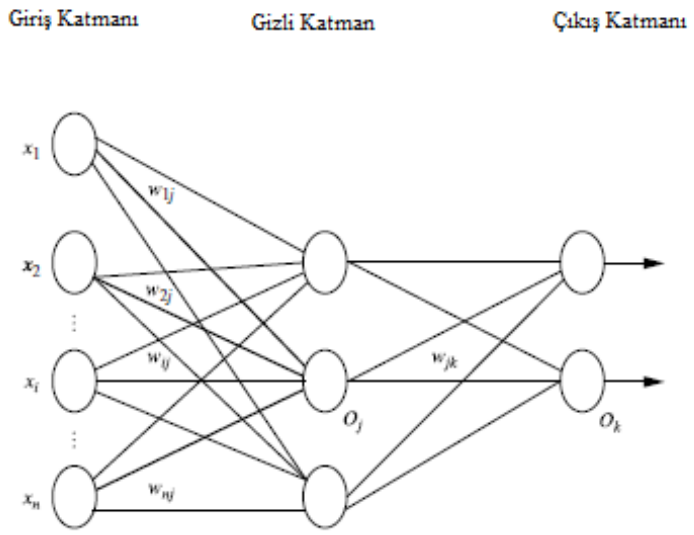
$$Chebyschev = \lim_{p\to\infty} \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}} = max_{i=1}^{n} |x_i - y_i| \quad (3.7)$$

3.2.3. Yapay sinir ağları

Yapay sinir ağları, insan beyninin özelliklerinin taklit edilmesiyle ve biyolojik sinir sisteminin simülasyon çalışmalarıyla insan beyninin işleyişinin matematiksel olarak modellemesini içeren bir öğrenme yöntemidir. İnsan beyninin temel elemanlarından birisi nöron adı verilen sinir hücreleridir. Sinir hücreleri hücre gövdesi, aksonlar ve dendritlerden oluşmaktadır. Nöronlar uyarıldığında sinir hücresi tarafından işlenen bilginin bir nörondan diğerine iletilmesi için aksonlar kullanılmaktadır. Bir nörondan başka bir nöronun aksonlarına bağlantı sağlanması için dendrit adı verilen hücre gövdesi uzantıları kullanılmaktadır. Sinyaller dendritler ile alındıktan sonra sinaps olarak adlandırılan iki nöron arasında bulunan temas noktasına ulaşıncaya dek akson üzerinden ilerlemektedir [34]. Yapay sinir ağları insan beynine benzer şekilde birbirine bağlı yapay sinir hücresi ve bağlantılardan oluşmaktadır. Yapay sinir ağına ait bilgi yapay sinir ağlarını birbirine bağlayan bağlantılar üzerindeki ağırlıklarda bulunmaktadır. Dışarıdan alınan bilgiler birleştirme fonksiyonu olarak da bilinen bir toplama fonksiyonu ile toplandıktan sonra aktivasyon fonksiyonundan geçirilmektedir. Aktivasyon fonksiyonu, toplama (birleştirme) fonksiyonu ile hesaplanan net girdi sinir hücresine girdikten sonra girdinin işlenmesini ve bir çıktı üretilmesini sağlamaktadır [36].

Birbirine paralel katmanlardan oluşan yapay sinir ağına girdi katmanından iletilen bilgi, ağırlık değerleri kullanılarak ara katmanlarda işlendikten sonra çıktı elde edilmektedir. Kullanılacak uygun ağırlıkların hesaplanması ağın eğitilmesiyle gerçekleşmektedir. Veri seti eğitim ve test seti olarak ayrıldıktan sonra eğitim aşamasının başlangıcında ağırlıklar rastgele atanmaktadır. Ağa gelen her örnek ile ağırlıklar öğrenme kuralına göre değiştirilmekte ve doğru değerler bulunmaya çalışılmaktadır. Eğitim örnekleri için doğru çıktılar elde edildikten sonra test setindeki örnekler için doğru yanıtlar alındığında ağ eğitilmiş olarak değerlendirilmektedir. Ağ eğitilirken "λ" sembolü ile gösterilen öğrenme katsayısı ve "α" sembolü ile gösterilen momentum katsayıları yapay sinir ağının öğrenme performansını etkilemektedir. Ağırlıkların değişim miktarları öğrenme katsayısı ile oluşturulmaktadır. Momentum katsayısı ise ağırlık değişiminin belirlenen bir oranının sonraki değişim miktarına eklenmesini ifade etmektedir [36, 37]. Yapay sinir ağları tek katmanlı ve çok katmanlı olabilmektedir. Yapay sinir ağlarının temelindeki ilk çalışmalar problem uzayının doğru/düzlem vasıtası ile sınıflara ayrılabildiği tek katmanlı algılayıcılardan oluşmaktadır. Tek katmanlı algılayıcılar sadece

giriş ve çıkış katmanlarından oluşmaktadır. Tek katmanlı algılayıcılarda ağ eğitilirken ağırlık değişimi girdi değerlerinin öğrenme katsayısı (λ) sabiti ile çarpılması ve ağırlıklara eklenip çıkarılması ile gerçekleşmektedir [36]. Tek katmalı algılayıcılarda doğrusal olmayan problemler için öğrenme sağlanmamaktadır. Bu problemin ortadan kaldırılabilmesi için çok katmanlı algılayıcılar kullanılmaktadır. Çok katmanlı algılayıcılarda dışarıdan gelen bilgilerin toplandığı giriş katmanı, girdi katmanından gelen bilgilerin işlendiği gizli katmanlar ve çıkış katmanından oluşmaktadır. Doğrusal olmayan problemler çok katmanlı algılayıcı ile çözülebilmektedir. Çok katmanlı algılayıcıya ait görsel Şekil 3.1'de verilmiştir [33].

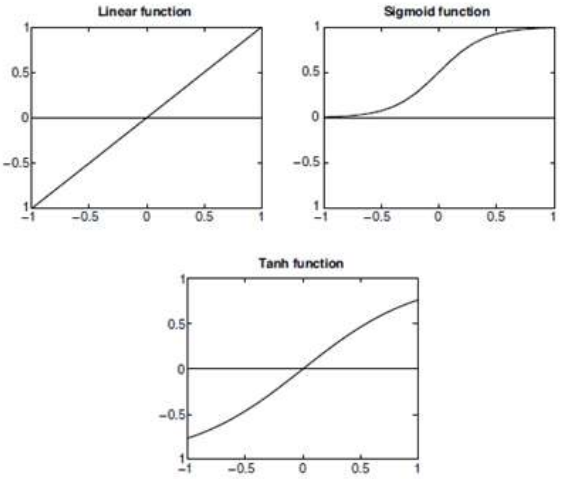


Şekil 3.1. *Çok katmanlı ileri beslemeli yapay sinir ağı*

Yapay sinir ağları yapılarına göre ileri beslemeli ve geri beslemeli yapay sinir ağları olarak ayrılmaktadırlar. Yapay sinir ağlarında doğrusal(lineer), sigmoid, hiperbolik tanjant, adım fonksiyonu gibi çeşitli aktivasyon fonksiyonları bulunmaktadır. Bu aktivasyon fonksiyonları, gizli katman ve çıkış katmanının doğrusal olmayan çıktı değerleri üretmesine izin vermektedir [34].

Doğrusal aktivasyon fonksiyonu, doğrusal problemlerin çözülmesi amacıyla seçilebilmektedir. Birleştirme fonksiyonundan elde edilen sonuç bir katsayıyla çarpıldığında çıkış değeri hesaplanabilmektedir. Sigmoid fonksiyonu doğrusal olmayan yapısı, sürekli ve türevlenebilir olma özellikleri ile yapay sinir ağı çalışmalarında en çok

kullanılan fonksiyon olmuştur. Girdi değerlerinden 0 ile 1 arasında sonuçlar alınmaktadır. Sigmoid fonksiyonu ile benzerlik gösteren hiperbolik tanjant fonksiyonunda çıkış değerleri –1 ile +1 arasında elde edilmektedir. Adım fonksiyonunda ise bilgiler 0 veya 0'dan küçük ise 0; 1 veya 1'den büyük ise 1 çıktısı elde edilmektedir [37].



Şekil 3.2. *Aktivasyon fonksiyonları*

Sinir hücresine gelen ağırlıklar ile girdi değerlerinin çarpımlarının toplamı ile sinir hücresine ait net girdi (3.8)'de gösterilen toplama fonksiyonu ile hesaplanmaktadır [37].

$$I_j = \sum_i w_{ij}\ O_i + \theta_j \tag{3.8}$$

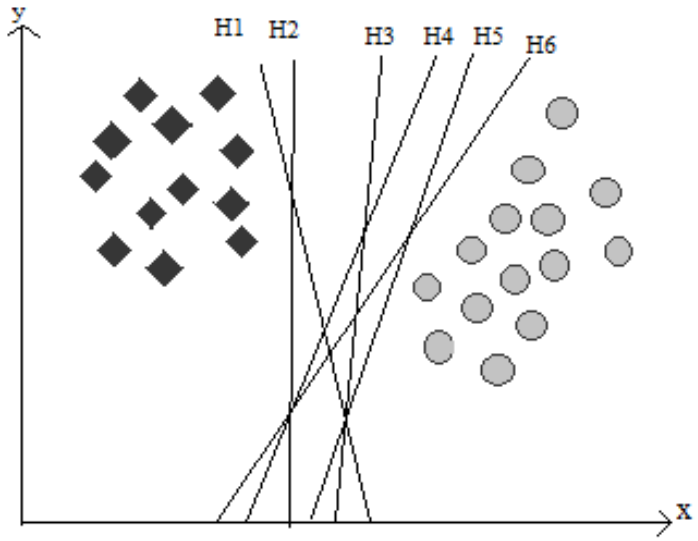
Burada Oi girdi değerini, wij i-j düğümleri arasındaki ağırlıkları, Ij toplama fonksiyonunu, θj eşik değerini (bias) temsil etmektedir.

(3.9)'da doğrusal (lineer), adım(step), sigmoid fonksiyonu ve hiperbolik tanjant aktivasyon fonksiyonlarına yer verilmiştir [33, 37]:

$$\begin{aligned} Doğrusal\ Fonksiyon&: F(I_j) = A \times I_j \\ Adım(Step)Fonksiyonu&: F(I_j) = \begin{cases} 1\ eğer\ I_j > \theta_j \\ 0\ eğer\ I_j \leq \theta_j \end{cases} \\ Sigmoid\ Fonksiyon&: F(I_j) = \frac{1}{1+e^{-I_j}} \\ Hiperbolik\ Tanjant\ Fonksiyonu&: F(I_j) = \frac{e^{I_j} + e^{-I_j}}{e^{I_j} - e^{-I_j}} \end{aligned} \tag{3.9}$$

3.2.4. Destek vektör makineleri

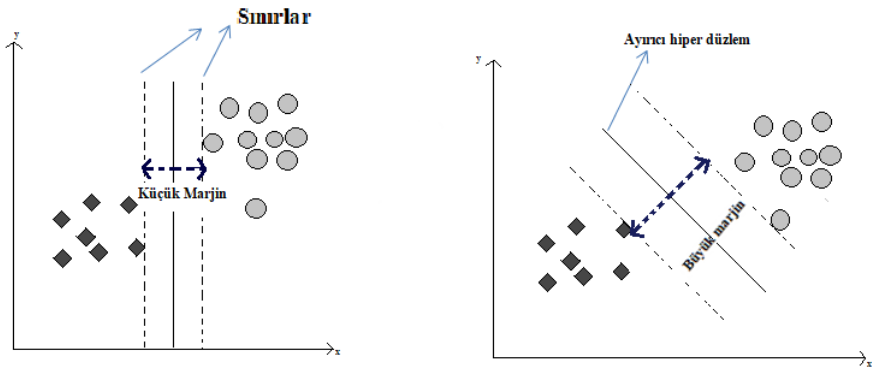
Destek vektör makineleri istatistiksel temele dayanan hem doğrusal olarak ayrılabilen verilerin hem de doğrusal olarak ayrılamayan verilerin sınıflandırılması için kullanılabilen bir yöntemdir. İki sınıfa ait verilerin ayrılabilmesi için en uygun düzlem ya da hiper düzlemin bulunması amaçlanmaktadır [33, 38]. Doğrusal olarak ayrılabilen veriler bir düzlem ile ayrılırken doğrusal olarak ayrılamayan veriler bulundukları boyuttan yüksek boyutlu uzaya taşınarak hiper düzlem ile sınıflara ayrılabilmektedir [38]. Şekil 3.3'te iki sınıfa doğrusal olarak ayrılabilen bir veri setine ait görsel verilmiştir.



Şekil 3.3. *Doğrusal olarak iki sınıfa ayrılabilen veriler*

Veri setini 2 sınıfa ayırabilmek için sonsuz sayıda doğru çizilebilmektedir. Destek vektör makineleri ile sınıflandırmada amaç N boyutlu uzayda sınıflandırma hatasını minimum yapacak maksimum marjinli hiper düzlemin belirlenmesidir [33, 38]. Bu amaca ulaşılabilmesi için ayırıcı hiper düzlem ile farklı iki sınıftan en yakın iki örneğin arasındaki uzaklığın maksimum olması gerekmektedir. Ayırıcı hiper düzleme en yakın olan farklı iki sınıfa ait örnekler destek vektörlerini oluşturmaktadır. Ayırıcı hiper düzlem iki örneğe eşit uzaklıkta bulunmaktadır. Optimum ayırıcı hiper düzlemin belirlenebilmesi için farklı sınıflara ait en yakın iki örnek arasındaki sınırı oluşturan, ayırıcı hiper düzleme paralel iki hiper düzlem kullanılması gerekmektedir. Destek vektörleri sınırı oluşturan hiper düzlemler üzerinde bulunmaktadır. Paralel iki hiper düzlem arasındaki mesafe

marjin olarak adlandırılmaktadır [33]. Doğrusal olarak ayrılabilen verilerin bulunduğu bir sınıflandırmada marjin ve sınırlara ait görsel Şekil 3.4'te verilmiştir.



Şekil 3.4. *Ayırıcı hiper düzlem ve sınırlar*

Hiper düzleme ait formül şu şekildedir:

$$< w.x > +b = 0 \tag{3.10}$$

Burada w (ağırlık vektörü) hiper düzlemin normalini, b yanlılığı, x hiper düzlem üzerinde seçilen rastgele bir noktayı göstermektedir. $< w.x >$ iç çarpımı $w^t x$ şeklinde yazıldığında hiper düzleme ait formül (3.11)'deki gibi olmaktadır.

$$w^t x + b = 0 \tag{3.11}$$

Destek vektörlerinin ayırıcı düzleme uzaklığı $\frac{1}{\|w\|}$ formülü ile, marjin uzunluğu $\frac{2}{\|w\|}$ formülü ile hesaplanmaktadır. Hesaplamalarda kullanılan ‖w‖, w'nin öklid normudur. Veri setinde bulunan n adet öznitelik için girdi vektörü xi (i=1, 2,…,n) ve sınıf etiketi $y_i \in \{-1,1\}$ ile ifade edildiğinde veri setindeki örneklerin en büyük marjine sahip hiper düzlemin ile ayrılabilmesi için sağlanması gereken eşitsizlikler şu şekildedir [38]:

$$w^t x_i + b \geq 1 \qquad y_i = +1 \ için \tag{3.12}$$

$$w^t x_i + b \leq 1 \qquad y_i = -1 \ için \tag{3.13}$$

(3.12) ve (3.13) eşitsizlikleri birleştirildiğinde (3.14)'teki eşitsizlik elde edilmektedir.

$$y_i(w^t x_i + b) \geq 1 \tag{3.14}$$

Maksimum marjinli ayırıcı hiper düzlemin bulunabilmesi için $\frac{2}{\|w\|}$ ifadesinde paydanın küçük olması istenmektedir. Bu durumda (3.15)'teki optimizasyon probleminin çözülmesi gerekmektedir.

$$Enk\ \frac{1}{2}\ \|w\|^2$$
$$y_i(w^t x_i + b) \geq 1\ \ \forall_i\ \ k.a. \tag{3.15}$$

İkinci dereceden eşitsizlik kısıtlı doğrusal olmayan optimizasyon probleminin çözümü için Lagrange çarpanlarından yararlanılmaktadır. Oluşturulan primal model (3.16)'da verilmiştir.

$$L_p = \frac{1}{2} w^t w - \sum_{i=1}^{n} \alpha_i y_i (w^t x_i + b) + \sum_{i=1}^{n} \alpha_i \tag{3.16}$$

Denklemde görülen $\alpha_i \geq 0$ olmak üzere problemin çözümünde kullanılan α_i değerleri Lagrange çarpanlarını oluşturmaktadır. Fonksiyonun w ve b'ye göre kısmi türevleri alınarak elde edilen Karush-Kuhn-Tucker koşulları şu şekildedir:

$$\sum \alpha_i y_i = 0 \tag{3.17}$$

$$w = \sum \alpha_i\, y_i x_i \tag{3.18}$$

Optimizasyon probleminin çözümü için oluşturulan dual model (3.19)'da verilmiştir.

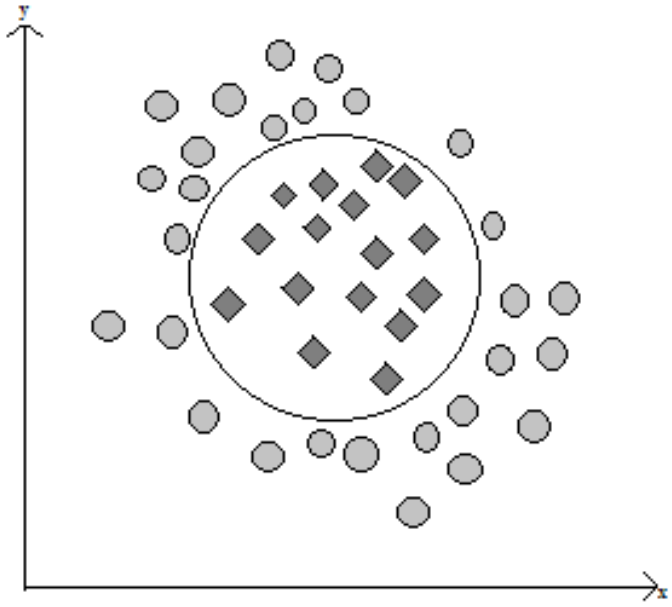
$$Enb\ L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^t x_j$$
$$\sum_{i=1}^{n} \alpha_i y_i = 0 \qquad \alpha_i \geq 0\ \ \forall_i \qquad k.a. \tag{3.19}$$

Sınıflandırma işleminin yapılması için kullanılabilecek fonksiyon şu şekildedir:

$$f(x) = (\sum \alpha_i y_i x_i\, .x) + b \tag{3.20}$$

Gerçek hayat problemlerinde çoğu zaman verilerin tamamının doğrusal olarak ayrılabilmesi mümkün olmamaktadır. Bu problemlerin çözülebilmesi için doğrusal olmayan haritalama tekniğinden faydalanılmaktadır. Bu teknikte veri seti daha yüksek boyutlu bir özellik uzayına taşınmaktadır. Burada maksimum marjinli ayırıcı hiper düzlemin çizilebilmesi için doğrusal bir ayrım yapılabilmektedir. Fakat veri setinin

bulunduğu asıl girdi uzayına iz düşümü doğrusal olmamaktadır [39]. Şekil 3.5'te doğrusal olarak ayrılamayan bir veri setine ait örnek verilmiştir.



Şekil 3.5. *Doğrusal olarak ayrılamayan bir veri setinin sınıflandırılması*

Destek vektör makineleri ile doğrusal olmayan bir veri setinin sınıflandırmasının yapılabilmesi için x'in daha yüksek boyutlu bir özellik vektörü olan Φ'ya dönüşümü yapılmaktadır. Bu işlem için Φ(x) fonksiyonu kullanılmaktadır. Bu durumda fonksiyon üzerinde dönüşüm yapıldığında elde edilen fonksiyon şu şekildedir [38]:

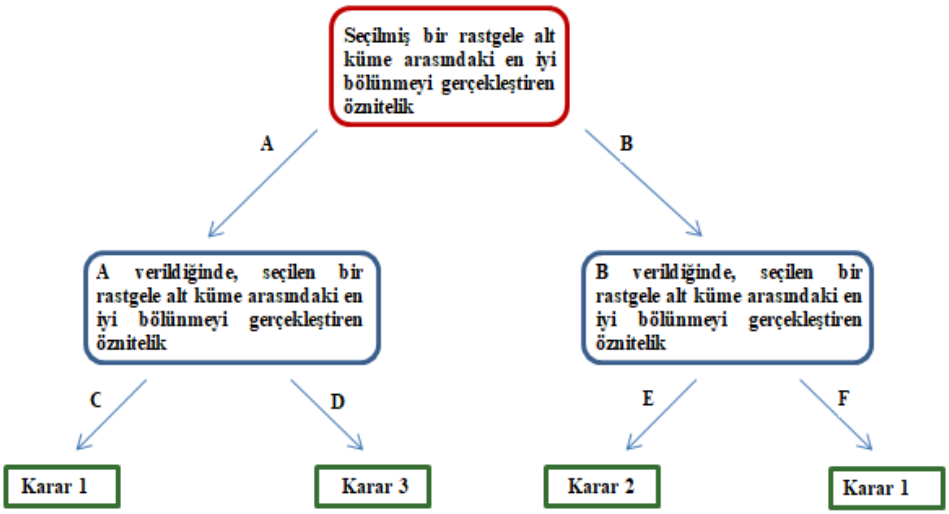
$$f(x) = \left(\sum \alpha_i \, y_i \Phi(x_i{}^t).\, \Phi(x_j)\right) + b \tag{3.21}$$

Φ(x) fonksiyonu ile hesaplama zorlularının ortadan kaldırılabilmesi için $\Phi(x_j{}^t).\,\Phi(x_j)$ iç çarpımı yerine çekirdek fonksiyonları kullanılabilmektedir. En çok kullanılan çekirdek fonksiyonları (3.22)'de yer almaktadır [33].

$$\begin{gathered} Polinomial\ Fonksiyon: K(x_i, x_j) = (x_i^T x_j + 1)^h \\ Sigmoid\ Fonksiyon: K(x_i, x_j) = \tanh\,(k x_i^t x_j - \delta) \\ Radyal\ Tabanlı\ Fonksiyon: K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}} \end{gathered} \tag{3.22}$$

3.2.5. Rastgele ağaç algoritması

Bir karar ağacı girdisi olmayan kök düğümü, öznitelikler ile ilgili testleri gösteren iç düğümler ve sınıf etiketini gösteren yaprak düğümlerden oluşmaktadır. Test sonuçları dallar üzerinde bulunmaktadır. Ağaçlardaki her bir iç düğüm özniteliklerden birisine karşılık gelmektedir. İç düğümlerde öznitelik değerlerine uygulanan testler sonucunda örnek uzay iki veya daha çok parçaya ayrılmaktadır. Karar ağacı algoritmalarından rastgele ağaç algoritması sınıflandırma kullanılabilen denetimli bir öğrenme algoritmasıdır [33, 40]. Standart bir karar ağacında her düğüm için tüm değişkenler arasında mümkün olan en iyi bölünme hesaplanırken rastgele ağaç yönteminde her düğümde tüm özniteliklerin rastgele seçilen bir alt kümesi dikkate alınmakta ve bu alt kümedeki öznitelikler kullanılarak en iyi bölünme hesaplanmaktadır. Bu yöntem hem sınıflandırma hem de regresyon problemleri üzerinde uygulanabilmektedir [41]. Rastgele ağaç yöntemine göre oluşturulan ağaç yapısı Şekil 3.6'da verilmiştir.



Şekil 3.6. *Rastgele ağaç yapısı*

Breiman tarafından önerilen rastgele ağaç yönteminde budama yapılmadan maksimum boyutlu ağaç üretilmesi için sınıflandırma ve regresyon ağacı (CART) metodolojisi kullanılmaktadır [42]. Bu metodolojide bölünmenin gerçekleşeceği özniteliklerin belirlenmesi için Gini endeksi kullanılmaktadır. Gini endeksi şu şekilde ifade edilmektedir [33]:

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2 \tag{3.23}$$

Verilen denklemde D eğitim veri setini, $|D|$ D'deki örnek sayısını, C_i (i=1,…,m) m ayrı sınıfı tanımlayan sınıf etiketini, $C_{i,D}$ D'deki C_i sınıfına ait örneklerin kümesini, $|C_{i,D}|$ $C_{i,D}$ 'deki örnek sayısını göstermektedir. Ci sınıfına ait örneklerin eğitim örneklerine oranı ile p_i (3.24)'teki gibi hesaplanmaktadır.

$$p_i = \frac{|C_{i,D}|}{|D|} \tag{3.24}$$

Her öznitelik için, olası ikili bölünmelerin her biri dikkate alınmaktadır. B özniteliği için ikili bölünme D_1 ve D_2 olarak gerçekleşirse bu öznitelik için Gini endeksi aşağıdaki gibi hesaplanmaktadır:

$$Gini_B^D = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \tag{3.25}$$

Kategorik değerli bir öznitelik için minimum Gini endeksini veren alt küme, bölme alt kümesi olarak seçilmektedir. Sürekli değere sahip özniteliklerde her bir sıralı komşu değer çifti için orta nokta olası bölme noktası olarak değerlendirilmektedir. Belirli bir sürekli değerli öznitelik için minimum Gini endeksini gösteren nokta, o özniteliğin bölme noktası olarak alınmaktadır. Kesikli veya sürekli değerli bir B özniteliğindeki ikili bölünmeden kaynaklanacak safsızlıktaki (heterojenlik) azalmaya ait denklem (3.26)'daki gibidir [33]:

$$\Delta Gini(B) = Gini(D) - Gini_B(D) \tag{3.26}$$

Minimum Gini endeksine sahip olan öznitelik, bölme özniteliği olarak seçilmektedir. Bu öznitelik ve bölme alt kümesi (kesikli değerli öznitelik için) veya bölme noktası (sürekli değerli öznitelik için) birlikte bölme kriterini oluşturmaktadır [33].

3.2.6. Öznitelik / bağımsız değişkenlerin seçimi

Öznitelik seçimi, daha verimli ve anlaşılabilir sonuçlar elde edilmesi için kullanılan bir veri boyutu azaltma tekniğidir. Yanıt değişkenin tahmin edilmesinde kullanılacak bağımsız değişkenler öznitelikleri oluşturmaktadır. Veri setinin yapısına göre bazı öznitelikler sonuç üzerinde etkisiz olabilir veya diğer özniteliklerden daha az etkiye sahip olabilir. Sınıflandırma algoritmalarında daha kolay yorumlama yapmak ve tahmin başarısını arttırmak için öznitelik seçimi yöntemlerinden yararlanılabilmektedir. Temel Bileşenler Analizi (TBA) ve Açıklayıcı Faktör Analizi (AFA) yöntemleri öznitelik seçimi

için bilinen ve yaygın kullanılan yöntemlerdendir. PROC VARCLUS Yöntemi, TBA ve AFA gibi geleneksel çok değişkenli yöntemlere alternatif olarak kullanılabilecek bir değişken azaltma tekniğidir. Bu yöntemde öznitelikler kümelere ayrılmakta ve küme bileşenleri tarafından açıklanan toplam varyansın yüksek olması amaçlanmaktadır. Regresyon analizinde de toplam varyasyonu en yüksek yapan ve en büyük etkiye sahip bağımsız değişkenlerin seçimi amaçlanmaktadır. Bir karşılaştırma yapılacak olursa PROC VARCLUS yöntemi, öznitelikleri kümeleme yaklaşımını benimseyen bir denetimsiz öğrenme tekniğidir. Regresyon ise modelin türetilmesi için bir sonucun kullanıldığı denetimli bir öğrenme tekniğidir. Çoğu veri madenciliği tekniği denetimli öğrenme tekniklerinden oluşsa da veri setinde gereksiz öznitelik olması durumunda öznitelik sayısının azaltılmasında denetimsiz öğrenme teknikleri kullanışlı olmaktadır [43].

3.2.7. Temel bileşenler analizi

Temel Bileşenler Analizi (TBA), çok sayıda birbiriyle ilişkili bağımsız değişkeni ve gereksiz değişkenleri de bulunduran bir grup veride gözlemlenen değişkenlerdeki varyansın çoğunu, temel bileşenler adı verilen daha az sayıda değişken ile açıklamaya yardımcı olan bir değişken azaltma metodudur [44].

Temel bileşenler cebirsel olarak n tane rassal değişkenin (X_1, X_2, X_3,…,X_n) doğrusal kombinasyonu ile oluşturulmaktadır. Geometriksel açıdan bu doğrusal kombinasyonlar orijinal sistemin eksenlerinin döndürülmesi ile değişkenliği en çok açıklayan ve kovaryans yapısını daha basit tanımlayan yeni eksenler elde edilmesini ifade etmektedir. Temel bileşenler rassal değişkenlerin "∑" sembolü ile gösterilen varyans kovaryans matrisinden elde edilmektedir [45]. Varyans kovaryans matrisinin özdeğer ve özvektör çiftleri (λ_1, e_1), (λ_2, e_2), (λ_3, e_3),…, (λ_n, e_n) olarak gösterilirse ve $\lambda_1 \geq \lambda_{2\ldots} \geq \lambda_n \geq 0$ olmak üzere, 1. Temel bileşen (Y_1) ve i. Temel bileşen (Y_i) şu şekilde hesaplanmaktadır:

$$Y_1 = e_1'X = e_{11}x_1 + e_{12}x_2 + \cdots + e_{1n}x_n \quad i = 1, 2, \ldots, n \tag{3.27}$$

$$Y_i = e_i'X = e_{i1}x_1 + e_{i2}x_2 + \cdots + e_{in}x_n \quad i = 1, 2, \ldots, n \tag{3.28}$$

Bu durumda Varyans ve kovaryans şu şekilde ifade edilmektedir [45]:

$$\mathrm{Var}(Y_i) = \lambda_i \quad i=1, 2,\ldots, n \tag{3.29}$$

$$\mathrm{Cov}(\mathrm{Y}_i, \mathrm{Y}_k) = 0 \quad \mathrm{i} \neq \mathrm{k} \tag{3.30}$$

Burada $\mathrm{Var}(\mathrm{Y}_i)$ varyansı, $\mathrm{Cov}(\mathrm{Y}_i, \mathrm{Y}_k)$ kovaryansı göstermektedir.

Birinci temel bileşen, değişkenliği en çok açıklayan orijinal değişkenlerin ağırlıklı doğrusal kombinasyonudur. İkinci temel bileşen, ilk temel bileşen ile ilişkisiz olan ve olası tüm ilişkisiz doğrusal kombinasyonlar arasındaki en fazla değişkenliği açıklayan ağırlıklı doğrusal kombinasyondur. TBA ile oluşturulan yeni değişkenlerin yorumlanmasının zorluğu bu yöntemin bir dezavantajı olarak kabul edilebilir. Ancak bu değişkenler döndürülerek aynı miktarda varyansı açıklayan, anlaşılması ve yorumlaması daha kolay olan değişkenler elde edilebilmektedir [44].

3.2.8. Açıklayıcı faktör analizi

Açıklayıcı faktör analizi (AFA) TBA ile yakından ilişkili bir değişken azaltma tekniğidir. Birçok yönden TBA ile benzerlik gösterse de kavramsal olarak bu iki yöntem arasında önemli farklılıklar bulunmaktadır. AFA daha çok değişkenlerin olduğu veri setindeki gizli faktörleri ve faktör yapılarını belirlemeye odaklanmıştır [44]. Ortak faktör modeline dayanan AFA'da veri kümesi içindeki her değişken bir veya birden fazla ortak faktörün ve bir özgün faktörün doğrusal kombinasyonudur. Ortak Faktörler değişkenlerin olduğu veri grubundaki birden fazla değişkeni etkileyen ve değişkenler arasındaki korelasyonları açıkladığı kabul edilen gözlemlenemeyen gizli değişkenlerdir. Özgün faktörler ise veri grubundaki sadece bir değişkeni etkileyen ve değişkenler arasındaki korelasyonu açıklamayan faktörlerdir. AFA'nın amacı ortak faktörler ile her bir ölçülen değişken arasındaki bağlantıyı tahmin ederek ortaya çıkarılan az sayıda gizli faktör ile değişkenler arasındaki korelasyonu anlamaktır. TBA'da ise toplam varyans göz önüne alınmakadır. Matematiksel olarak temel bileşenler veri grubundaki orijinal değişkenin doğrusal kombinasyonları olarak tanımlanabildiğinden hem ortak varyansı hem de özgün varyansı içerebilmektedirler. Aynı sayıda temel bileşen ile faktör seçildiğinde farklı yöntemlerle elde edilen farklı tiplerdeki faktör ve temel bileşen genel olarak benzer sonuçlar vermektedir. Ancak daha kolay hesaplanabilir oluşu ve daha az bellek kullanımı gerektirmesi TBA'yı avantajlı hale getirmektedir [46, 47].

3.2.9. PROC VARCLUS yöntemi

PROC VARCLUS Yöntemi sadece sürekli değişkenler kullanılarak uygulanabilen ve iterasyona dayanan bir kümeleme yöntemidir. Bu yöntem ile oluşturulacak küme sayısını daha önceden belirlemek mümkün olabileceği gibi algoritmanın durdurma kriteri sağlandığında oluşan küme sayısını kullanmak da değerlendirilebilir bir seçenektir. Aşağıdaki kriterlerden biri sağlandığında algoritma kümelere bölmeyi sonlandırmaktadır [48]:

- Maksimum küme sayısı belirlenmiş ise küme sayısı belirlenen sayıdan büyük ya da eşit olduğunda,
- Varyans açıklanma oranı durdurma kriteri olarak belirlenen bir oranı sağladığında veya ikinci özdeğer belirlenen bir eşik değerinden düşük olduğunda algoritma sonlanır.

PROC VARCLUS Yöntemi'nin varsayılan uygulamasında her kümede ikinci öz değerin 1'den küçük olması durdurma kriteri olarak kabul edilmektedir. Ancak çözülmek istenen probleme göre bu durum üzerinde değişiklik yapılabilmektedir. SAS adlı istatistiksel analiz yazılımında PROC VARCLUS yönteminin uygulandığı bir değişken kümeleme algoritması bulunmaktadır. Bu algoritma kullanılarak değişkenler kümelenmektedir. Küme bileşenleri ile veya kümeyi temsil edecek temsilci değişkenler seçilmişse bu değişkenler ile veri setindeki değişkenliğin çoğunluğu açıklanabilmektedir. Temsilci değişkenler daha sonra tahmin edici modellerde veya farklı modellerde kullanılabilmektedir [48, 49]. Algoritma başlangıcında tüm değişkenler tek bir kümenin içindedir.

PROC VARCLUS Yöntemi ile değişken kümeleme algoritmasının iteratif adımları aşağıdaki gibidir [48, 49]:

- Her kümedeki değişkenler için temel bileşenler hesaplanır. Tüm kümelerde 2.özdeğer 1'den küçükse algoritmayı sonlandırılır. (Başlangıçta tek bir küme vardır.)
- İkinci özdeğeri en büyük olan (ve 1'den büyük olan) küme geçerli küme olarak belirlenir ve 2 yeni kümeye aşağıdaki adımlar izlenerek bölünür:
  - Geçerli kümedeki temel bileşenler Harris ve Kaiser'in döndürme yöntemine göre döndürülür.

- Geçerli kümede birinci döndürülmüş temel bileşen ile kare korelasyonu (R^2) ikinci temel bileşenle kare korelasyonundan (R^2) yüksek olan mevcut kümedeki değişkenlerden oluşan bir küme tanımlanır.
- Orijinal kümede kalan diğer değişkenler için bir küme oluşturulur. Bu değişkenler ile 2. Temel bileşen arasında yüksek derecede korelasyon vardır.
- İki yeni kümenin temel bileşenlerini hesaplanır.

- Veri kümesindeki herhangi bir değişkenin farklı bir kümeye atanıp atanmayacağı test edilir. İzleyen maddeler her değişken için uygulanır:
  - Her kümede değişkenler ile birinci temel bileşenler arasındaki kare korelasyon (R^2) hesaplanır.
  - Değişken kare korelasyonunun (R^2) en büyük olduğu kümeye yerleştirilir.

TBA ile PROC VARCLUS aralarında birtakım benzerlikler ve farklılıklar bulunmaktadır. TBA'da veri setindeki tüm değişkenler analize dâhil edilmektedir ve temel bileşenler bu değişkenlerin doğrusal kombinasyonu ile oluşturulmaktadır. PROC VARCLUS yönteminde ise tüm değişkenlerin olduğu veri seti kümelere ayrılmaktadır. Her kümede ilk temel bileşen kullanılarak küme bileşenleri oluşturulmaktadır. Yöntemin varsayılan uygulamasında ve genel kullanımda ilk temel bileşenler küme bileşenlerini oluştursa da ağırlık merkezi bileşenleri de küme bileşeni olarak seçilebilmektedir. Öznitelik seçiminde değişkenliği en çok açıklayan değişkenler küme temsilcileri olarak seçilebilmektedir. PROC VARCLUS ile değişken kümeleme yapılmasının dezavantajlarından birisi kategorik yapıdaki değişkenlerin analize dâhil edilememesidir. Yöntemin uygulanabilmesi için değişkenlerin sadece sürekli ve numerik değer alması gerekmektedir. PROC VARCLUS yönteminin istatistiksel bir yazılımda uygulanan bir algoritma olarak başlaması, çok değişkenli yöntemler hakkındaki standart ders kitaplarında veya bilimsel dergilerde uzunca açıklanan algoritma olmamasının yöntemin uygulanmasına karşı önyargı oluşturduğu söylenebilir. Aslında geleneksel yöntemlere göre birçok avantajı bulunan bu yöntemin uygun problemlerde kullanılması ile başarılı sonuçlar elde edilebilir [44].

AFA'da kullanılan faktörler ve TBA'da kullanılan temel bileşenler orijinal değişken sayısından daha az sayıda fakat orijinal değişkenlerin bilgileri kullanılarak elde edilen yeni değişkenleri oluşturmaktadır. Bu iki yöntemde çalışmanın devamında yapılacak analizlerde yeni oluşturulan değişkenler kullanılmakta iken PROC VARCLUS yönteminde öznitelik seçimi sonrasında kullanılan değişkenler orijinal değişkenlerden oluşmaktadır.

PROC VARCLUS yöntemi ile değişken kümelemenin avantajlarından birisi de bu yöntemin yüzlerce değişkene sahip bir veri setinde hızlı bir şekilde ve kolaylıkla uygulanabilmesidir. PROC VARCLUS yöntemi ile kolay yorumlanabilir kararlı modeller etkin bir şekilde oluşturulabilmektedir. Kullanılan algoritma ile modelleme süreci de dikkate değer ölçüde hızlandırılabilmektedir. Bu yöntem ile oluşturulan istatistiksel tabloların yorumlanması uzmanlık gerektirmediğinden analizi yapan kişi tarafından alternatif küme temsilcileri kolayca ortaya çıkarılabilmekte ve önemsiz bulunan girdi değişkenleri modelden atılabilmektedir. TBA gibi diğer değişken azaltma yöntemlerinde bu şekilde yorumlanabilir kümeler oluşturulamamaktadır [43, 48, 49, 50].

4. DİYABET

Diyabet, kandaki glikoz seviyesi yükseldiğinde pankreastan yeterli miktarda insülin hormonu salgılanamaması sonucu ortaya çıkan ve yaşam boyu devam eden bir hastalıktır. Enerji ihtiyacının karşılanması için karbonhidrat içeren çoğu besin glikoza dönüştürülmektedir. Midenin arka yüzeyinde bulunan pankreas organı kasların ve diğer dokuların kandaki glikozu enerji olarak kullanabilmelerini sağlayan insülin hormonunu üretmektedir. Bu hormon, besinler yoluyla kana geçen glikozun yakıt olarak kullanılmak üzere hücrelere girmesini sağlamaktadır. İnsülin, protein ve yağ metabolizması için de gereklidir. Glikozun ihtiyaçtan daha fazla olması durumunda fazla olan miktar karaciğerde yağ olarak depolanmaktadır [3, 51].

Diyabetli olmayan bir insanın açlık kan şekerinin en fazla 120 mg/dl, tokluk kan şekerinin ise en fazla 140 mg/dl olması beklenmektedir. Açlık ve tokluk kan şekerlerinin bu değerlerden yüksek olması diyabet hastalığının göstergesidir. Diyabetin tespit edilebilmesi için açlık kan şekeri ölçümü veya oral glikoz tolerans testi yapılabilmektedir. Açlık kan şekerinin 100-125 mg/dl aralığında olması gizli şeker olarak da bilinen pre-diyabetin, 126 mg/dl veya daha fazla olması diyabetin varlığına işaret etmektedir. Oral glikoz tolerans testine kişiye belirli bir düzeyde glikoz çözeltisi verildikten sonra belirli aralıklarla kandaki glikoz düzeyi takip edilmektedir. 2 saat sonra kan şekeri ölçümü yapıldığında değerin 200 mg/dl ve üzerinde çıkması diyabetin göstergesidir [51].

4.1.Diyabet Tipleri

Diyabet hastalığının Tip I diyabet, Tip II diyabet, gebelikte görülen gestasyonel diyabet ve monogenetik diyabet olarak adlandırılan farklı çeşitleri bulunmaktadır.

4.1.1. Tip I diyabet

Tip 1 diyabet, vücudun bağışıklık sisteminin insülin üreten beta hücrelerine saldırdığı bir otoimmün reaksiyondan kaynaklanmaktadır. Sonuç olarak vücutta yeterli miktarda insülin hormonu üretilemediğinden insülin eksikliği görülmektedir. Tip 1 diyabeti olan kişiler, kan şekeri seviyelerini korumak için günlük insülin takviyesine gereksinim duymaktadır. Tip 1 diyabet çocuklar ve gençlerde daha sık görülse de her yaşta görülebilmektedir. Sebebi tam olarak anlaşılamamış olsa da, genetik yatkınlığı olan

bir bireyde viral enfeksiyon gibi çevresel bir tetikleyici otoimmün reaksiyonu başlatmaktadır. Tip I diyabette görülen belirtiler aşırı susuzluk, bulanık görme, idrara çıkmada sıklık, yorgunluk, halsizlik, sürekli açlık hissi ve ani kilo kaybı şeklinde olmaktadır. Tip 1 diyabet teşhisi, belirtilerin bazılarının veya tümünün varlığında yüksek kan şekerinin tespiti ile konmaktadır fakat Tip 1-Tip II ve özellikle monogenetik tipler arasında ayrım yapılmasında zorlanıldığı durumlarda ek testler gerekebilmektedir [3].

4.1.2. Tip II diyabet

Tip II diyabet, en sık görülen diyabet tipidir. Dünya üzerinde teşhis edilen diyabet vakalarının %90'ını oluşturmaktadır. Vücut hücrelerinin insüline tam olarak yanıt verememesi sonucunda insülin direnci oluşmakta ve insülin üretimi yetersiz kalmaktadır. Ailede diyabet öyküsü bulunması, aşırı kilo, sağlıksız beslenme, yüksek tansiyon, artan yaş, fiziksel hareketsizlik gibi faktörler Tip II diyabetle ilişkili risk faktörleridir. Tip II diyabet sıklıkla yaşlı yetişkinlerde görülse de artan obezite prevelansı, fiziksel hareketsizlik ve uygun olmayan beslenme nedeniyle çocuklarda ve genç yetişkinlerde görülme sıklığı giderek artmaktadır. Tip II diyabet, Tip I diyabet ile benzer belirtilerle ortaya çıkabilse de belirtiler daha hafif olmakta veya tamamen belirtisiz de olabilmektedir. Tip II diyabetin başlangıç zamanının belirlenmesi daha zordur. Uzun bir süre belirlenemediğinde komplikasyonlar görülebilmektedir. Tip II diyabetin yaş ve genetik faktörlerin yanı sıra aşırı kilo ve obezite ile de arasında güçlü bir ilişki olduğu bilinmektedir. Kandaki glikoz seviyelerinin, kan basıncının ve kan lipit seviyelerinin kontrolü hayati önem taşımaktadır. Metabolik faaliyetlerin düzenli olarak takip edilmesi ile görülebilecek komplikasyonların gelişimi izlenebilmektedir. Düzenli kontroller, yaşam tarzının etkili bir şekilde yönetilmesi ve gerekirse ilaç tedavisi uygulanması ile Tip II diyabet hastaları uzun ve sağlıklı bir yaşam sürdürebilmektedir [3, 51].

4.1.3. Gestasyonel diyabet

Hamilelik döneminde tespit edilen gestasyonel diyabet hamileliğin herhangi bir döneminde ortaya çıkabilmektedir. Bu diyabet türünde hamilelik döneminde pankreasın yeterli düzeyde insülin salgılayamaması sonucunda kan şekeri yüksekliği görülmektedir. Gestasyonel diyabet prevalansı yaşa bağlı olarak artmaktadır. 45 yaş ve üzeri hamilelerde

risk en fazladır. Gestasyonel diyabeti olan kadınlarda yüksek tansiyon, büyük doğum ağırlıklı bebekler ve engelli doğum gibi hamilelikle ilgili komplikasyonlar yaşanmaktadır. Hamilelik sırasında diyabet öyküsü olan kadınların yaklaşık yarısında doğum sonrasındaki 5-10 yıl içinde Tip II diyabet görülmektedir [3, 52].

4.1.4. Monogenetik diyabet

Monogenetik diyabet Tip I ve Tip II diyabette görülen çoklu genler ve çevresel faktörlerin katkılarından ziyade tek bir genden kaynaklanmaktadır. Bebeklik, çocukluk ve 25 yaş altı genç yetişkinlerde görülen bu diyabet türünde bozuk olan gen belirlenerek tanı konulmaktadır [53]. Tüm diyabet vakalarının %1,5- %2'sini oluşturduğu söylense de gerçekte bu oranın daha yüksek olduğu düşünülmektedir. Çünkü monogenetik diyabet türü Tip I ve Tip II diyabet ile karıştırılmakta ve yanlış teşhis edilebilmektedir. Bazı durumlarda tedavi belirli bir genetik bozukluğa göre uygulandığından diyabetin monogenetik formlarının kesin teşhisi önemlidir [3].

4.2.Diyabetin Komplikasyonları

Diyabetli insanlarda diyabetin yanı sıra ciddi sağlık problemleri gelişebilmektedir. Kan şekeri seviyesinin sürekli yüksek olması kalbi, kan damarlarını, gözleri, böbrekleri, sinirleri ve dişleri etkileyen ciddi hastalıklara sebep olmaktadır. Yüksek kan şekeri olan kişilerde enfeksiyon gelişme ihtimali de daha yüksektir. Yüksek gelirli ülkelerin çoğunda kardiyovasküler hastalık, körlük, böbrek yetmezliği ve alt ekstremite ampütasyonunun önde gelen nedeni diyabettir [54].

5. DİYABET HASTALIĞI TEŞHİSİNDE KÜMELEME VE SINIFLANDIRMA ALGORİTMALARININ KULLANILDIĞI BİR KARAR DESTEK SİSTEMİNİN GELİŞTİRİLMESİ

Bu çalışmada Adana'da hizmet veren bir sağlık kuruluşunda diyabeti olan ve diyabeti olmayan hastalardan dosya taraması yoluyla alınan bilgilerden oluşan veri seti kullanılmıştır. Veri seti sınıflandırma işlemleri öncesinde hazırlık aşamasından geçirilmiştir. Veri setinin kullanıma uygun hale getirilebilmesi için ön işleme tekniklerinden yararlanılmıştır. Veri madenciliği sürecinin gerçekleştirilmesinde aşağıdaki adımlar izlenmiştir:

- *Veri temizleme:* Çalışmada kullanılmasına karar verilen veri setinde veri setinde bulunan eksik ve hatalı verilerin ortadan kaldırılabilmesi için veri temizleme işlemi gerçekleştirilmiştir.
- *Veri dönüştürme ve boyut azaltma:* Veri setindeki öznitelikler standardize edilmiştir. Z skoru standardizasyonu uygulanmıştır. Boyut azaltma için PROC VARCLUS değişken kümeleme tekniğinden faydalanılmış ve elde edilen kümelerden küme temsilcileri seçilmiştir.
- *Veri Madenciliği Tekniklerinin Uygulanması:* Çalışmada kullanılmasına karar verilen Naive Bayes, Yapay sinir ağı (çok katmanlı algılayıcı), destek vektör makineleri, k en yakın komşu ve karar ağacı sınıflandırma algoritmaları uygulanmıştır.
- *Model değerlendirme:* Belirlenen değerlendirme kriterlerine göre sınıflandırma algoritmalarından elde edilen sınıflandırma başarıları değerlendirilmiştir.
- *Karma bir sınıflandırıcı önerisi:* Sınıflandırma algoritmaları birlikte kullanılarak çoğunluğun tahmin sonucuna göre karma bir sınıflandırıcı önerilmiştir.
- *Karar destek sistemi geliştirilmesi:* Çalışmada kullanılan sınıflandırma algoritmaları ve geliştirilen karma yönteme ait tahmin sonuçlarına kullanıcı arayüzü vasıtasıyla hekimler tarafından kolaylıkla ulaşılabilmesi için bir karar destek sistemi geliştirilmiştir.

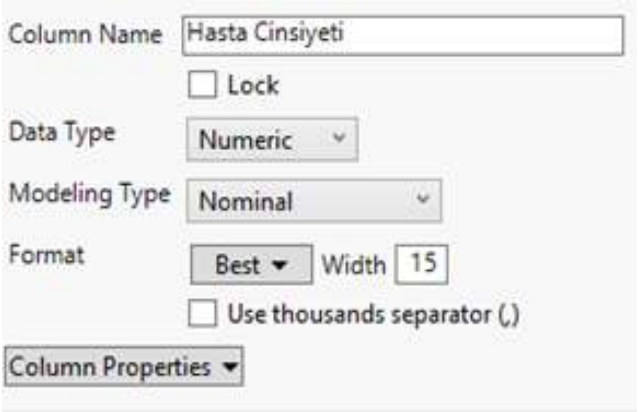
5.1. Değişken Kümeleme Tekniği ile Öznitelik Seçimi Yapılması

Öznitelik seçimi ile değişken sayısının azaltılması için SAS adlı istatiksel veri analiz yazılımından yararlanılmıştır. Öznitelik seçim yöntemi olarak, ileri düzey bir uzmanlık gerektirmeden, kolay yorumlanabilir tabloların oluşturulmasına olanak sağladığından ve memnun edici sonuçlar verdiğinden PROC VARCLUS yöntemi kullanılmıştır. PROC VARCLUS yöntemi ile kümelenen değişkenler arasında dâhil olduğu kümedeki varyansı en çok açıklayan değişkenler hastalık teşhisi için küme temsilcileri olarak seçilmiştir. Veri setinde 52'si diyabeti olan, 56'sı diyabeti olmayan olmak üzere 108 hastaya ait 17 değişken bulunmaktadır. Bu değişkenler ve değişkenlere ait kısa açıklamalar Tablo 5.1'de gösterilmiştir.

Tablo 5.1. *Çalışmada kullanılan değişkenler ve açıklamaları*

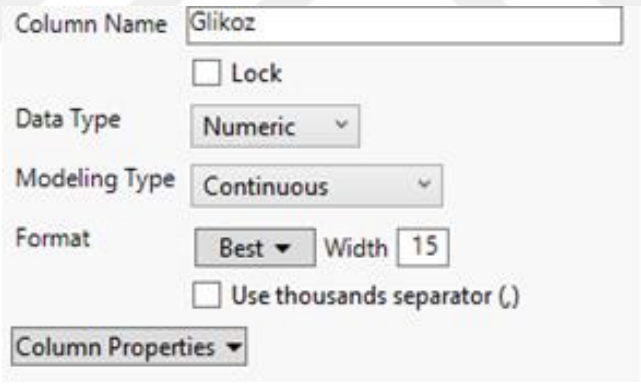
Değişken Adı	Açıklama
Hasta Cinsiyeti	0: Kadın, 1: Erkek
Yaş	Çalışmaya dâhil edilen kişinin yaşı
Glikoz	Kan Şekeri
Kolesterol	Kan plazmasında bir yağ türü
LDL Kolesterol	Kolesterol taşınmasında görevli düşük yoğunluklu lipoprotein
HDL Kolesterol	Kolesterol taşınmasında görevli yüksek yoğunluklu lipoprotein
Trigliserit	Kanda bulunan ve vücudun enerji ihtiyacında kullanılan bir yağ türü
RBC	Kırmızı kan hücresi
HGB	Hemoglobin, kırmızı kan hücresi proteini
HCT	Kırmızı kan hücrelerinin kandaki oranı
MCV	Ortalama hücre hacmi
MCH	Kırmızı kan hücresinde ortalama hemoglobin miktarı
MCHC	Kırmızı kan hücresinde ortalama hemoglobin konsantrasyonu
PDW	Trombosit dağılım genişliği
EOS%	Eozinofil yüzdesi
EOS#	Eozinofil sayısı
Hastalık Durumu	0: Diyabet hastalığı yok 1: Diyabet hastalığı var

Cinsiyet değişkeni ve hastalık durumu değişkeni kategorik yapıdadır. Cinsiyet değişkeni için 0: Kadın, 1: Erkek cinsiyetleri temsil etmektedir. Tahmin edilmeye çalışılan hastalık durumu değişkeni iki sınıflıdır. 0: "Diyabet hastalığı yok", 1:" Diyabet hastalığı var" anlamına gelmektedir. Kategorik değişkenler için sütun bilgisi kısmından veri tipi ve model tipi seçenekleri değişkenlerin türüne uygun olacak şekilde yeniden düzenlenmiştir. Şekil 5.1'de hasta cinsiyetine ait sütun bilgisi gösterilmiştir. Kategorik yapıdaki cinsiyet değişkenine uygun model tipi seçilmiştir. Aynı düzenleme kategorik yapıda olan hastalık durumu değişkeni için de yapılmıştır.



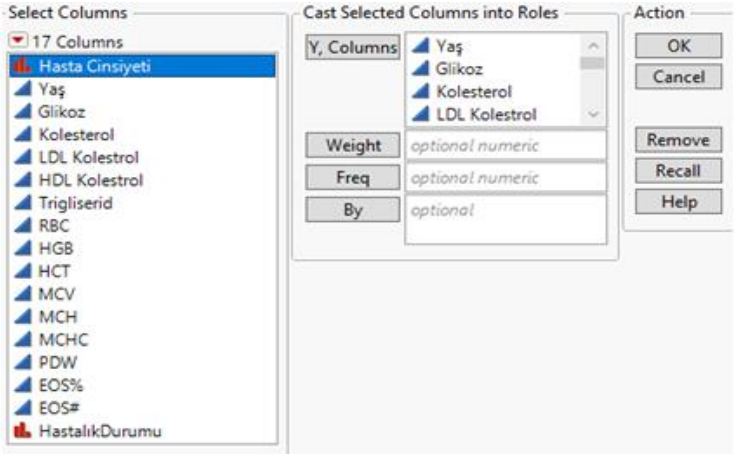
Şekil 5.1. *Hasta cinsiyeti değişkeninin sütun bilgisi*

Kalan 15 değişken sürekli yapıdadır. Şekil 5.2'de glikoz değişkenine ait sütun bilgisi gösterilmiştir. Glikoz değişkeninin aldığı değerler sürekli yapıda olduğundan model tipi sürekli olarak belirlenmiştir. Kategorik değişkenler haricindeki tüm değişkenlerin sütun bilgileri benzer şekilde oluşturulmuştur.



Şekil 5.2. *Glikoz değişkeninin sütun bilgisi*

Şekil 5.3'te algoritmada kullanılacak değişkenlerin seçiminin yapıldığı sayfa gösterilmiştir.



Şekil 5.3. *Algoritmada kullanılacak değişkenler*

Sadece sürekli yapıdaki değişkenler için uygun olduğundan bu değişkenler seçilerek algoritma çalıştırılmıştır. 15 değişken, değişken kümeleme algoritması ile 5 kümeye ayrılmıştır. Her bir kümenin üyeleri, küme bileşenleri ile küme üyeleri arasında hesaplanan korelasyon katsayılarının kareleri Tablo 5.2'de gösterilmiştir.

Tablo 5.2. *Değişkenler için hesaplanan R^2 değerleri*

Cluster	Members	RSquare with Own Cluster(R^2_{own})	RSquare with Next Closest ($R^2_{nearest}$)	1-RSquare Ratio
1	HCT	0,903	0,095	0,108
1	RBC	0,85	0,136	0,173
1	HGB	0,816	0,091	0,202
1	Yaş	0,185	0,026	0,836
2	MCH	0,932	0,035	0,07
2	MCV	0,871	0,023	0,132
2	MCHC	0,09	0,01	0,919
3	Trigliserit	0,601	0,117	0,452
3	Glikoz	0,513	0,036	0,506
3	HDL Kolesterol	0,417	0,049	0,614
3	PDW	0,397	0,101	0,67
4	EOS%	0,948	0,012	0,053
4	EOS#	0,948	0,015	0,053
5	LDL Kolesterol	0,893	0,015	0,109
5	Kolesterol	0,893	0,077	0,116

1-R^2, bir kümenin ait olduğu küme ile en yakın olduğu kümenin arasındaki göreceli yakınlığın ölçüsüdür [49]. Formülü (5.1)'de verilmiştir.

$$1 - R^2 = (1 - R^2_{own})/(1 - R^2_{nearest} \quad (5.1)$$

Verilen denklemde R^2_{own} değişkenin bulunduğu kümedeki küme bileşeni ile arasındaki korelasyon katsayısının karesini, $R^2_{nearest}$ ise değişkenin tüm küme bileşenleri ile arasındaki korelasyon katsayısının karesinin ikinci en büyük değerini ifade etmektedir. Her küme üyesinin kendi ait olduğu kümenin küme bileşeni ile korelasyonu diğer küme bileşenleriyle korelasyonundan mümkün olduğunca daha yüksek olması istendiğinden R^2_{own} değerinin yüksek olması beklenmektedir. $R^2_{nearest}$ değerinin ise düşük olması istenmektedir. Bu değerin düşük olması kümelere ayırmanın başarılı bir şekilde yapıldığını göstermektedir [48]. 1-R^2_{own} değerinin düşük, 1-$R^2_{nearest}$ değerinin yüksek bir değer olması istendiğinden (5.1) eşitliğinin sağ tarafının düşük bir değer olması beklenmektedir.

Tablo 5.2'de 3. sütunda her değişkenin bulunduğu kümedeki küme bileşeni ile arasındaki korelasyon katsayısının karesi (R^2_{own}) hesaplanmıştır. Değişkenlerin R^2_{own} değerleri yüksekten düşüğe doğru olacak şekilde sıralanmıştır. En yüksek R^2_{own} değerine sahip değişken bulunduğu kümedeki varyansı en çok açıklayan değişkendir. 4. sütunda ise sonraki en yakın R^2 değerleri bulunmaktadır. $R^2_{nearest}$ değerleri her bir değişken ile diğer küme bileşenleri arasındaki R^2 değerlerinin hesaplanması ve 2. en yüksek olan değerin seçilmesi ile elde edilmiştir. 5.sütunda 1-R^2 oranı hesaplanmıştır. Bu çalışmada her bir kümedeki en düşük 1-R^2 oranına sahip değişkenin kendi küme bileşeni ile arasındaki R^2_{own}değeri 0,90 ve üzerindeyse tek bir temsilci, daha düşükse en düşük 1-R^2 oranına sahip 2 temsilci seçilmiştir. Öznitelik seçiminin sınıflandırma başarısını arttırdığı görülmüştür. Sınıflandırma algoritmalarında kullanılmak üzere seçilen değişkenlerin olduğu satırlar Tablo 5.2'de koyu renkle işaretlenmiştir.

Tüm kümelerdeki toplam üye sayısı m, bir kümedeki üye sayısı n adet olduğunda kümenin açıklanan varyansı, varyans açıklama oranı, toplam varyans açıklama oranı şu şekilde hesaplanmaktadır:

$$Variation\ Explained = \sum_{i=1}^{n} R^2_{i\,own} \tag{5.2}$$

$$Proportion\ of\ Variation\ Explained = \frac{\sum_{i=1}^{n} R^2_{i\,own}}{n} \tag{5.3}$$

$$Total\ Proportion\ of\ Variation\ Explained = \frac{\sum_{i=1}^{m} R^2_{i\,own}}{m} \tag{5.4}$$

Tablo 5.2'deki küme elemanlarının R^2_{own} değerlerinin toplanmasıyla her bir kümedeki açıklanan varyans hesaplanmıştır. Açıklanan varyanslar Tablo 5.3'te 4.sütunda

gösterilmiştir. Tablo 5.3'te her bir kümedeki açıklanan varyansın kümedeki üye sayısına bölünmesiyle varyans açıklama oranı hesaplanmıştır. Varyans açıklama oranı 5.sütunda gösterilmiştir. Her bir küme için açıklanan varyansın toplam üye sayısına bölünmesi ile kümelerin toplam varyans açıklama oranları hesaplanmıştır. Bu oranlar Tablo 5.3'te 6.sütunda gösterilmiştir. Kümeler, toplam varyans açıklanma oranı büyükten küçüğe doğru olacak şekilde sıralanmıştır. Bir örnekle açıklanacak olursa 1 numaralı küme 4 değişkenden oluşmaktadır. HCT, RBC, HGB, yaş değişkenlerinin R^2_{own} değerleri toplamı 2.754 olarak bulunmuştur. Bu değer 4'e bölündüğünde kümenin varyans açıklama oranı olan 0.689 değeri elde edilmiştir. Toplam varyans açıklama oranı ise R^2_{own} değerleri toplamının 15'e bölünmesiyle 0.184 olarak hesaplanmıştır. Diğer kümelere ait hesaplamalar da benzer şekilde yapılmıştır.

Tablo 5.3. *Açıklanan varyanslar*

Cluster	Members	Cluster Variation	Variation Explained	Proportion of Variation Explained	Total Proportion of Variation Explained
1	4	4	2,754	0,689	0,184
3	4	4	1,928	0,482	0,129
4	2	2	1,896	0,948	0,126
2	3	3	1,893	0,631	0,126
5	2	2	1,786	0,893	0,119

Standart bileşenler her kümedeki birinci temel bileşen dikkate alınarak hesaplanmıştır. Her değişken yalnızca bir kümeye atandığından her satırda sadece bir tane değerin sıfırdan farklı olduğu görülmektedir. Tablo 5.4'te belirtilen katsayılar kümelerdeki ilk temel bileşenlerin özvektörlerini oluşturmaktadır.

Tablo 5.4. *Standartlaştırılmış bileşenler*

Standardized Components					
Variable	**Cluster 1 Coefficients**	**Cluster 2 Coefficients**	**Cluster 3 Coefficients**	**Cluster 4 Coefficients**	**Cluster 5 Coefficients**
Yaş	-0,259363	0	0	0	0
Glikoz	0	0	0.5156834	0	0
Kolesterol	0	0	0	0	0,7071068
LDL Kolesterol	0	0	0	0	0,7071068
HDL Kolesterol	0	0	-0,4649050	0	0
Trigliserid	0	0	0,5584352	0	0
RBC	0,5555934	0	0	0	0
HGB	0,5443779	0	0	0	0
HCT	0,5724505	0	0	0	0
MCV	0	0.6783309	0	0	0
MCH	0	0.7016636	0	0	0
MCHC	0	0.2180259	0	0	0
PDW	0	0	0,4539653	0	0
EOS%	0	0	0	0,7071068	0
EOS#	0	0	0	0,7071068	0

Kategorik yapıda olması sebebiyle değişken kümeleme algoritmasında yer almayan cinsiyet değişkeni için yapılan araştırma sonucunda 2018 Yılı Sağlık İstatistikleri Yıllığı'nda açıklanan oranlara göre Türkiye'de diyabet hastalığına sahip kadınların oranının erkeklerden daha yüksek olduğu görülmüştür. Ayrıca bu değişkenin hastalık teşhisi için kullanılan sınıflandırma algoritmalarında doğru sınıflandırma oranını artırdığı tespit edilmiştir. Öznitelik seçimi sonrası sınıflandırma algoritmalarında kullanılmasına karar verilen değişkenler, değişkenlerin türleri ve model tipi bilgileri Tablo 5.5'te gösterilmiştir.

Sürekli değişkenlere ek olarak kategorik yapıdaki cinsiyet değişkeni ve sınıflandırıcı olarak kabul edilen hastalık durumu değişkeni de WEKA'da sınıflandırma algoritmalarında kullanılmak üzere Tablo 5.5'e eklenmiştir.

Tablo 5.5. *Öznitelik seçimi ile seçilen değişkenlere ait bilgiler*

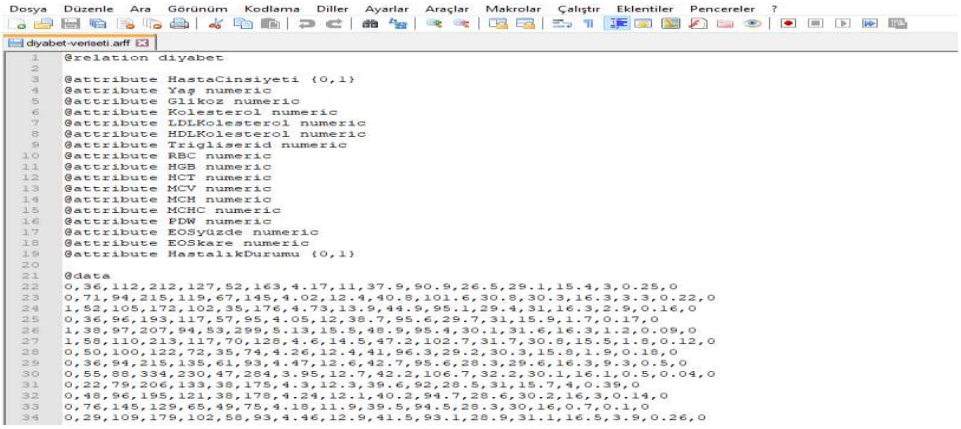
Değişken Adı	Model Tipi/Veri Tipi	Değişken Türü (Girdi-çıktı)
Hasta Cinsiyeti	Kesikli / Kategorik	Girdi
Glikoz	Sürekli /Nümerik	Girdi
Kolesterol	Sürekli /Nümerik	Girdi
LDL Kolesterol	Sürekli /Nümerik	Girdi
Trigliserit	Sürekli /Nümerik	Girdi
HCT	Sürekli /Nümerik	Girdi
MCH	Sürekli /Nümerik	Girdi
EOS%	Sürekli /Nümerik	Girdi
Hastalık Durumu	Kesikli / Kategorik	Çıktı

5.2. WEKA Veri Madenciliği Yazılımının Kullanılması

Çalışmada WEKA veri madenciliği yazılımından yararlanılmıştır. WEKA yazılımı java programlama dilini kullanan, öğrenme algoritmaları uygulamalarını içeren açık kaynaklı bir yazılımdır. Waikato Üniversitesi'nde 1992 yılında bir proje olarak başlamıştır. Kullanıcıların kaynak kodlarına ücretsiz erişim sağlayabilmesi projelerin yürütülmesini ve WEKA'nın gelişimini hızlandırmıştır. Günümüzde veri madenciliği araştırmalarında yaygın olarak kullanılmaktadır. Dosya, URL ve veri tabanları gibi çeşitli kaynaklardan programa veri yüklemek mümkündür. Arff, CSV, C4.5 gibi formatlar WEKA'da desteklenen dosya formatları arasındadır [55]. En çok kullanılan formatlar Arff ve CSV formatlarıdır. Bu çalışmada Xls formatındaki veriler Arff formatına çevrilmiştir. WEKA'da programın kurulumuyla birlikte hazır olarak gelen pek çok öğrenme algoritması bulunmaktadır. Hastalık teşhisi için kullanılabilecek sınıflandırma algoritmalarını içerdiğinden ve algoritmalardan alınan sonuçların görselleştirilmesine de olanak sağladığından bu yazılım tercih edilmiştir.

Xls formatındaki verilerin virgül ile ayrılması için dosya başlangıçta CSV dosyası olarak kaydedilmiştir. Daha sonra veriler Arff formatına uygun olacak şekilde yeniden düzenlenerek Arff uzantısı ile kaydedilmiştir. Arff formatında başlık ve veri olmak üzere 2 ayrı kısım bulunmaktadır. İlk kısım başlık kısmıdır. Başlık kısmında "@relation" ve "@attribute" ifadeleri bulunmaktadır. İlk satırda "@relation" ifadesi ile ilişki tanımı yapılmaktadır. İlişki tanımı veri hakkında bilgiler içermektedir. Eğer tanımlanan ilişkiyi açıklayan kelimelerin arasında boşluk bırakılıyorsa kelimelerin tırnak içinde yazılması

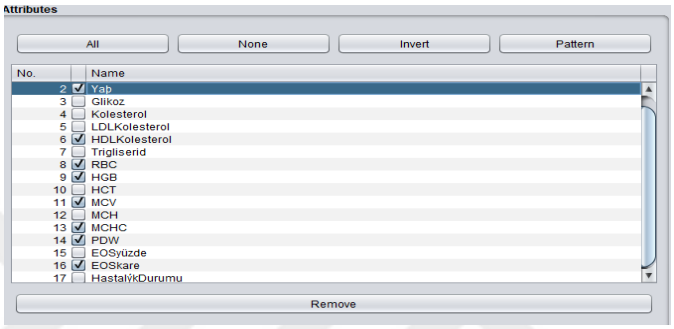
gerekmektedir. Diyabet hastalığının teşhisi için bir çalışma yapıldığından ilişki diyabet olarak tanımlanmıştır. Excel dosyasındaki her bir sütun ismi değişkenlerin isimlerini oluşturmaktadır. Değişkenler "@attribute" ifadesi ile tanımlanmaktadır. Değişkenlerin excel dosyasında bulundukları sütunların sıralarına göre tanımlanmaları gerekmektedir. Ayrıca değişkenin yapısına göre tipinin de belirtilmesi gerekmektedir. Sayısal değerler için 'numeric', reel sayılar için "real", metin bilgileri için "string" veri tipi seçimi yapılmaktadır. Veri kısmında "@data" ifadesinin altına ise her bir değişkenin her bir örnekte aldığı değerler virgül ile ayrılarak yazılmaktadır. Bu çalışmada sayısal değerler ile çalışıldığından değişken tipleri "numeric" olarak seçilmiştir. Arff dosyasında "@relation" ifadesi ile öncelikle ilişki belirtilmiştir. Diyabet hastalığının teşhisi için bir çalışma yapıldığından ilişki diyabet olarak tanımlanmıştır. Sınıflandırma algoritmalarında kullanılmasına karar verilen değişkenler "@attribute" ifadesi ile her bir değişken ayrı satırda olacak şekilde yazılmıştır. Değişken tipleri değişken adının yanında bir boşluk bırakılarak kaydedilmiştir. "@data" ifadesinin altına hastalara ait her bir hastanın verisi bir satırda olacak şekilde ve değişkenlerin aldığı değerler tanımlanma sırasına uygun olacak şekilde virgül ile ayrılarak yazılmıştır. Diyabet veri setinin Arff formatına göre düzenlenmiş hali Şekil 5.4'te gösterilmiştir.



Şekil 5.4. *Diyabet veri seti için düzenlenen Arff dosyası*

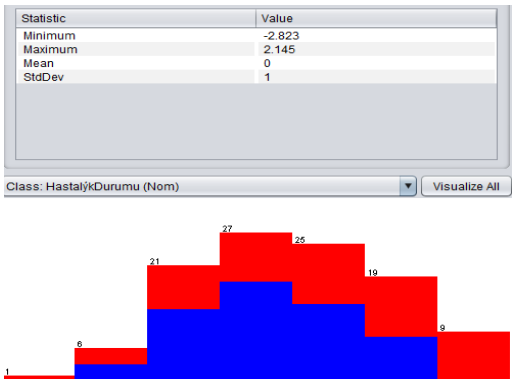
Bu çalışmada WEKA programının 3.9.3 versiyonu kullanılmıştır. Program çalıştırıldığında açılan arayüzde çeşitli uygulamalar bulunmaktadır. Hastalık teşhisi için sınıflandırma algoritmalarının bulunduğu "Explorer" uygulaması kullanılmıştır. WEKA'da sınıflandırma algoritmaları ile analiz yapılmadan önce ön işleme sürecinde

standardizasyon işlemi uygulanmıştır. Öznitelikler ortalaması 0, varyansı 1 olacak şekilde yeniden ölçeklendirilmiştir. Ön işleme sekmesi altında Arff formatında hazırlanan diyabet dosyası açılmıştır. Dosya açıldıktan sonra öznitelik seçimi sonrası sınıflandırma algoritmalarında kullanılmasına karar verilen Tablo 5.5'te belirtilen değişkenlerin dışında kalan değişkenler "Remove" butonu ile kaldırılmıştır. Bu işlem Şekil 5.5'te gösterilmiştir.



Şekil 5.5. *Seçilen değişkenlerin düzenlenmesi*

Değişkenlerin listelendiği sekmede her değişken için standart sapma, ortalama, en küçük, en büyük değerler görülebilmektedir. Görselleştirme seçeneği ile değişkenlerin aldığı değerlerin aralıklarına göre diyabet hastalığın dağılımının olduğu grafik görülebilmektedir. Şekil 5.6'da HCT değişkenine ait bilgiler ve dağılım grafiği gösterilmiştir. Mavi renk diyabet hastalığı olmayan kişileri, kırmızı renk diyabet hastalığı olan kişileri temsil etmektedir.



Şekil 5.6. *HCT değişkenine ait bilgiler ve dağılım grafiği*

Classify sekmesi ile açılan sayfada çalışılan veri seti eğitim ve test veri seti olarak ikiye ayrılmıştır. Veri setindeki verilerin %66'sı eğitim verisi, %34'ü test verisi olarak seçilmiştir.

5.3. Değerlendirme Ölçütleri

Diyabet hastalığının tespitinde sınıflandırma algoritmalarının sınıflandırma başarılarının hesaplanmasında kullanılacak değerlendirme ölçütleri için karmaşıklık matrisinden yararlanılmıştır. Değerlendirme ölçütlerinin hesaplanması için oluşturulan karmaşıklık matrisi Şekil 5.7'de verilmiştir. Karmaşıklık matrisi, tahmin edilen sınıf ve gerçek sınıf ile ilgili bilgiler içermektedir.

		Tahmin edilen sınıf	
		0 (Diyabet hastalığı yok)	**1 (Diyabet hastalığı var)**
Gerçek sınıf	**0 (Diyabet hastalığı yok)**	Gerçek Negatif (GN)	Yanlış Pozitif (YP)
	1 (Diyabet hastalığı var)	Yanlış Negatif (YN)	Gerçek Pozitif (GP)

Şekil 5.7. *Karmaşıklık matrisi*

Gerçek Pozitif (GP): Sınıflandırma işlemi sonucunda diyabet hastalığının olduğu tahmin edilen bir kişinin gerçekte de diyabet hastalığının var olduğu durumlardır.

Gerçek Negatif (GN): Sınıflandırma işlemi sonucunda diyabet hastalığının olmadığı tahmin edilen bir kişinin gerçekte de diyabet hastalığına sahip olmadığı durumlardır.

Yanlış Pozitif (YP): Sınıflandırma işlemi sonucunda diyabet hastalığının olduğu tahmin edilen bir kişinin gerçekte diyabet hastalığına sahip olmadığı durumlardır.

Yanlış Negatif (YN): Sınıflandırma işlemi sonucunda diyabet hastalığının olmadığı tahmin edilen bir kişinin aslında diyabet hastalığına sahip olduğu durumlardır.

Karmaşıklık matrisinden aşağıdaki değerlendirme ölçütlerinin hesaplanması mümkündür:

Doğru sınıflandırma oranı: Gerçek negatif ve gerçek pozitif örneklerin toplamının tüm örneklerin toplamına oranıdır. Doğru sınıflandırma oranının formülü (5.5)'te verilmiştir.

$$Doğru\ Sınıflandırma\ Oranı = \frac{GP+GN}{GP+YP+GN+YN} \tag{5.5}$$

Kesinlik: Pozitif olarak tahmin edilen ve gerçekte pozitif olan vakaların oranını göstermektedir [56]. Bu değer diyabet hastalığının olduğu tahmin edilen örneklerin ne kadarının doğru olduğunun bir ölçüsüdür. Yani diyabet hastası olarak tahmin edilen kişilerin ne kadarının gerçekten diyabet hastalığına sahip olduğunun ölçüsüdür. Kesinliğin formülü (5.6)'da gösterilmiştir.

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

Duyarlılık: Pozitif olarak doğru tahmin edilen gerçek vakaların oranıdır [56]. Gerçek pozitif sayısının gerçekten diyabet hastalığına sahip olan kişi sayısına oranı ile hesaplanmıştır. Duyarlılığın formülü (5.7)'de verilmiştir.

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

F ölçütü: Kesinlik ve duyarlılık değerlerinin harmonik ortalamasıdır. Formülü (5.8)'de gösterilmiştir [56].

$$F\ ölçütü = \frac{2\times Kesinlik\times Duyarlılık}{Kesinlik+Duyarlılık} \tag{5.8}$$

Kappa İstatistiği: Karmaşıklık matrisinden hesaplanabilecek ölçütlerden birisi de Kappa istatistiğidir. Tesadüfi ve şans faktörünü de hesaba katan, beklenen ve gözlenen doğruluk değerlerin karşılaştırılması esasına dayanan bir güvenilirlik ölçüsüdür. Tesadüfi bir sınıflandırıcının beklenen doğruluk değeri hesaba katılarak yapılan sınıflandırmanın gerçeğe ne kadar yakın sınıflandırıldığı ile ilgili bir değerlendirme yapılmasını sağlamaktadır. Kappa istatistiği (5.9)'da gösterilen formül ile hesaplanmaktadır [57].

$$Kappa\ istatistiği = \frac{GD-BD}{1-BD} \quad (5.9)$$

Burada GD gözlenen doğruluğu, BD beklenen doğruluğu ifade etmektedir.

Denklem (5.9)'daki formülden 0 ile 1 arasına bir değer elde edilmektedir. Elde edilen değerin yorumlanması hususunda kesin kabul görmüş standart bir yöntem olmasa da genel olarak 0,00 ile 0,60 aralığında hesaplanırsa orta, orta altı ve düşük; 0,61 ile 0,80 aralığında hesaplanırsa önemli; 0,81-1,00 aralığında mükemmel olarak değerlendirilmektedir [58].

5.4.Sınıflandırma Algoritmalarının Sonuçlarının Değerlendirilmesi

Bu çalışma kapsamında WEKA programında Bayes sekmesi altında bulunan sınıflandırma yöntemlerinden en iyi sonuç veren Naive Bayes algoritması; fonksiyonlar sekmesi altındaki sınıflandırma yöntemlerinden destek vektör makinesi ve çok katmanlı algılayıcı (yapay sinir ağı) algoritması; lazy sekmesi altındaki k en yakın komşu (KNN) algoritması; ağaçlar sekmesi altında bulunan rastgele orman, J48 ve rastgele ağaç sınıflandırıcılarından en iyi sonuç veren rastgele ağaç algoritması ele alınmıştır. Ele alınan algoritmaların değerlendirilmesinde doğru sınıflandırma oranı, kesinlik, duyarlılık, F ölçütü ve Kappa istatistiği değerleri kullanılmıştır.

WEKA programında öğrenme algoritmalarından alınan sınıflandırma sonuçları aşağıdaki gibidir:

- *Naive Bayes Algoritması*

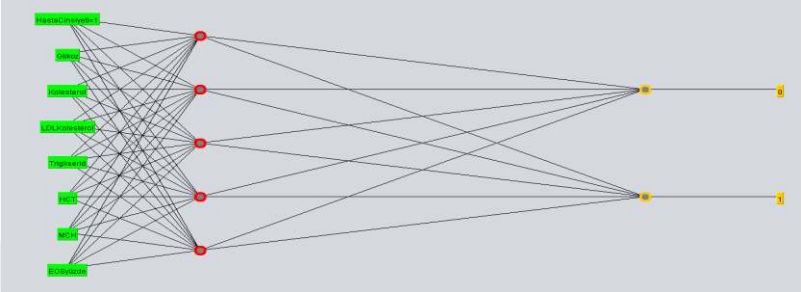
Naive Bayes algoritması ile yapılan sınıflandırmaya göre hesaplanan sonuçlar Tablo 5.6'da verilmiştir.

Tablo 5.6. *Naive Bayes algoritmasına göre sınıflandırma sonuçları*

Değerlendirme Ölçütü	**Sonuç**
Doğru Sınıflandırma Oranı (%)	%91,892
Kesinlik (Precision)	0,857
Duyarlılık (Recall)	1
F Ölçütü (F measure)	0,923
Kappa İstatistiği	0,838

➢ *Çok katmanlı algılayıcı (YSA)*

Çok katmanlı algılayıcı algoritmasına göre oluşturulan gizli katmanlı yapay sinir ağı Şekil 5.8'de gösterilmiştir. Yeşil renkler giriş katmanlarını, kırmızı renkler gizli katmanları, sarı renkler ise çıkış katmanlarını göstermektedir.



Şekil 5.8. *Gizli katmanlı yapay sinir ağı*

Seçilen değerlendirme ölçütleri için çok katmanlı algılayıcı algoritmasına göre hesaplanan sonuçlar Tablo 5.7'de verilmiştir.

Tablo 5.7. *Çok katmanlı algılayıcı algoritmasına göre sınıflandırma sonuçları*

Değerlendirme Ölçütü	**Sonuç**
Doğru Sınıflandırma Oranı (%)	%91,892
Kesinlik (Precision)	0,895
Duyarlılık (Recall)	0,944
F Ölçütü (F measure)	0,919
Kappa İstatistiği	0,838

➢ *Destek Vektör Makineleri (Sıralı Minimum Optimizasyon)*

WEKA programında Sıralı minimum optimizasyon algoritması (SMO) olarak adlandırılan destek vektör makinelerine ile yapılan sınıflandırmaya göre hesaplanan sonuçlar Tablo 5.8'de verilmiştir.

Tablo 5.8. *Destek vektör makineleri algoritmasına göre sınıflandırma sonuçları*

Değerlendirme Ölçütü	Sonuç
Doğru Sınıflandırma Oranı (%)	%81,081
Kesinlik (Precision)	0,824
Duyarlılık (Recall)	0,778
F Ölçütü (F measure)	0,800
Kappa İstatistiği	0,621

- *K-En Yakın Komşu (KNN)*

Değerlendirme ölçütleri için WEKA'da K-en yakın komşu algoritmasına göre hesaplanan sonuçlar Tablo 5.9'da verilmiştir. Hesaplamalarda çeşitli k değerleri denenmiştir. En yüksek doğru sınıflandırma oranı k=1 alındığında elde edildiğinden hesaplamalarda en yakın bir komşu kullanılmıştır.

Tablo 5.9. *K-En yakın komşu algoritmasına göre sınıflandırma sonuçları*

Değerlendirme Ölçütü	Sonuç
Doğru Sınıflandırma Oranı (%)	%81,081
Kesinlik (Precision)	0,867
Duyarlılık (Recall)	0,722
F Ölçütü (F measure)	0,788
Kappa İstatistiği	0,620

- *Rastgele Ağaç (Karar Ağacı)*

WEKA programında RandomTree olarak adlandırılan rastgele ağaç karar ağacı öğrenme algoritmasına göre oluşturulan karar ağacı Şekil 5.9'da gösterilmiştir.

```
RandomTree
==========

Trigliserit < -0.19
|   Glikoz < -0.27
|   |   MCH < -0.89
|   |   |   Kolesterol < -0.71 : 1
|   |   |   Kolesterol >= -0.71 : 0
|   |   MCH >= -0.89 : 0
|   Glikoz >= -0.27
|   |   LDLKolestrol < -0.89
|   |   |   Glikoz < 0.05 : 0
|   |   |   Glikoz >= 0.05 : 1
|   |   LDLKolestrol >= -0.89 : 1
Trigliserit >= -0.19
|   MCH < 0.24
|   |   LDLKolestrol < 1.22 : 1
|   |   LDLKolestrol >= 1.22 : 0
|   MCH >= 0.24
|   |   Glikoz < -0.53 : 0
|   |   Glikoz >= -0.53 : 1
```

Şekil 5.9. *Rastgele ağaç algoritmasına göre oluşturulan ağaç*

Değerlendirme ölçütleri için rastgele ağaç algoritmasına göre hesaplanan sonuçlar Tablo 5.10'da verilmiştir.

Tablo 5.10. *Rastgele ağaç (RandomTree) karar ağacı algoritmasına göre sınıflandırma sonuçları*

Değerlendirme Ölçütü	**Sonuç**
Doğru Sınıflandırma Oranı (%)	%83,784
Kesinlik (Precision)	0,773
Duyarlılık (Recall)	0,944
F Ölçütü (F measure)	0,850
Kappa İstatistiği	0,677

Sınıflandırma işlemi için kullanılan algoritmaların tamamı Tablo 5.11'de gösterilmiştir. Değerlendirme ölçütlerine göre en iyi sonuç veren 3 sınıflandırma algoritması koyu renk ile işaretlenmiştir.

Tablo 5.11. *Sınıflandırma işlemi için kullanılan algoritmalar*

Sınıflandırma Yöntemi	**Doğru Sınıflandırma Oranı (%)**	**Kesinlik (Precision)**	**Duyarlılık (Recall)**	**F Ölçütü (FMeasure)**	**Kappa İstatistiği**
Naive Bayes	%91,892	0,857	1	0,923	0,838
Yapay Sinir Ağı (Çok Katmanlı Algılayıcı)	%91,892	0,895	0,944	0,919	0,838
Destek Vektör Makinesi (SMO)	%81,081	0,824	0,778	0,800	0,621
K En Yakın Komşu (IBk)	%81,081	0,867	0,722	0,788	0,620
Karar Ağacı (RandomTree)	%83,784	0,773	0,944	0,850	0,677

Öznitelik seçiminin sınıflandırma üzerindeki etkisinin görülebilmesi için öznitelik seçimi öncesi [1] ve öznitelik seçimi sonrasında [2] sınıflandırma algoritmalarının sonuçları karşılaştırılmıştır. Karşılaştırma sonuçları Tablo 5.12'de verilmiştir.

Tablo 5.12. *Öznitelik seçimi öncesi ve sonrası değerlendirme ölçütleri için sınıflandırma algoritmalarından alınan sonuçlar*

Sınıflandırma Yöntemi	**Naive Bayes**		**Yapay Sinir Ağı (Çok Katmanlı Algılayıcı)**		**Destek Vektör Makinesi (SMO)**		**K En Yakın Komşu (IBk)**		**Karar Ağacı (RandomTree)**	
-	1	2	1	2	1	2	1	2	1	2
Doğru Sınıflandırma Oranı (%)	91,89	91,89	83,784	91,892	78,378	81,081	64,865	81,081	75,676	83,784
Kesinlik (Precision)	0,857	0,857	0,773	0,895	0,778	0,824	0,692	0,867	0,696	0,773
Duyarlılık (Recall)	1	1	0,944	0,944	0,778	0,778	0,500	0,722	0,889	0,944
F Ölçütü(F measure)	0,923	0,923	0,85	0,919	0,778	0,800	0,580	0,788	0,780	0,85
Kappa İstatistiği	0,838	0,838	0,677	0,838	0,567	0,621	0,292	0,620	0,517	0,677

*Öznitelik seçimi öncesi değerlendirme ölçütleri için hesaplanan sonuçlar [1] ile öznitelik seçimi sonrası hesaplanan sonuçlar [2] ile gösterilmiştir.

Tablo 5.12'deki sonuçlar değerlendirildiğinde yapay sinir ağı, destek vektör makinesi, k en yakın komşu ve karar ağacı algoritmaları için öznitelik seçimi sonrası doğru sınıflandırma oranının arttığı, Naive Bayes algoritması için ise değerlendirme ölçütlerinin değişmediği görülmüştür. K en yakın komşu ve karar ağacı algoritmaları için tüm değerlendirme ölçütleri artış göstermiştir. Yapay sinir ağı ve destek vektör makineleri için duyarlılık değeri değişmezken diğer değerlendirme ölçütleri artış göstermiştir. PROC VARCLUS Yöntemi ile öznitelik seçimi sonrasında genel olarak yapılan değerlendirme sonucunda değerlendirme ölçütleri için daha başarılı sonuçlar elde edildiği görülmüştür.

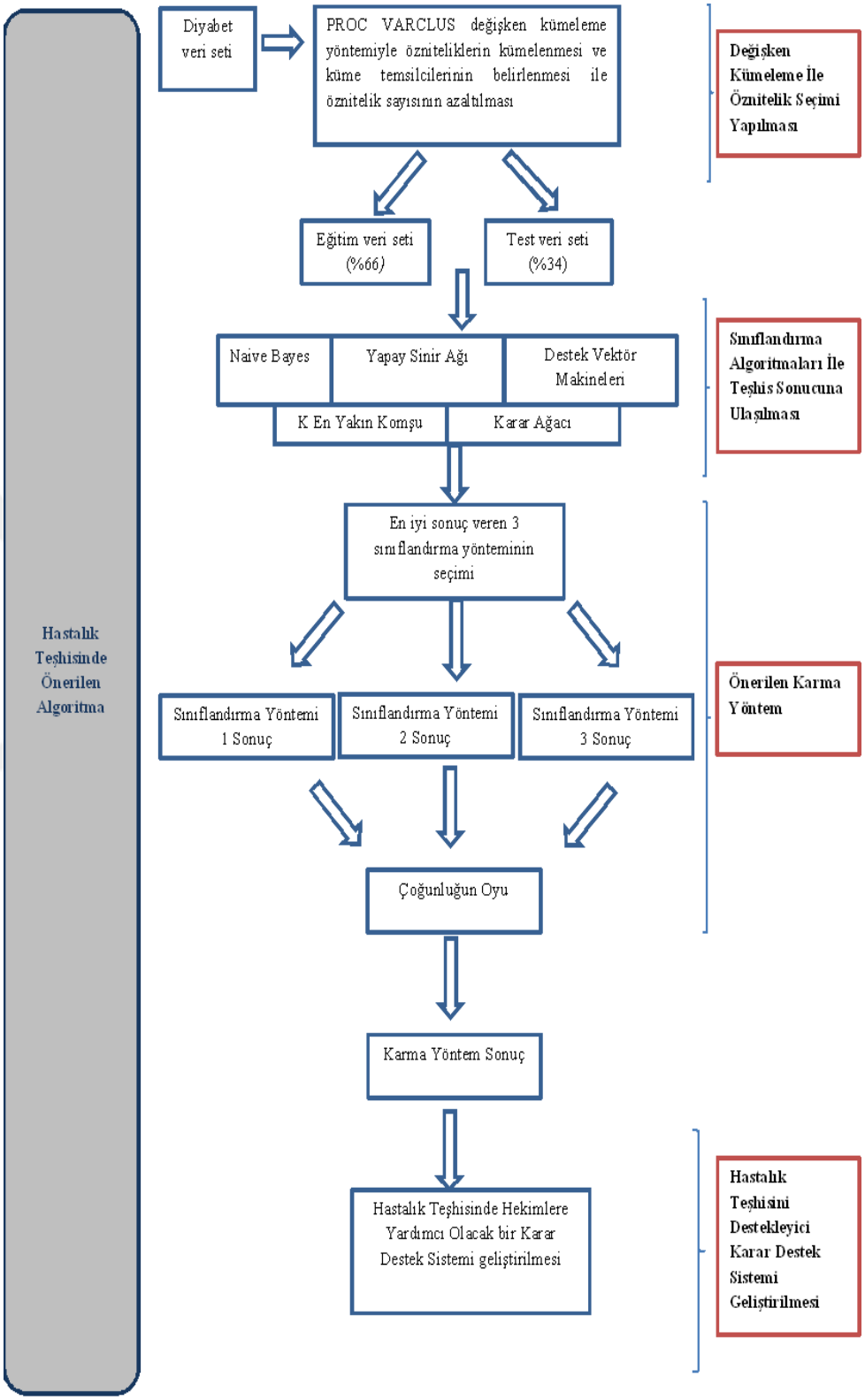
5.5. Sınıflandırma Algoritmalarının Birlikte Kullanılması ile Karma bir Yöntem Önerilmesi

Diyabet hastalığının teşhisi için doğru sınıflandırma oranı, kesinlik, duyarlılık, F ölçütü ve Kappa istatistiği değerlendirme ölçütlerine göre sınıflandırma algoritmaları karşılaştırıldığında en iyi sonuç veren üç yöntemin Naive Bayes, çok katmanlı algılayıcı (YSA) ve rastgele ağaç karar ağacı olduğu belirlenmiştir. Üç yöntemden alınan tahmin sonuçlarının birleştirilmesiyle karma bir karar verici yöntem oluşturulmuştur. 2 sınıflı bir veri seti ile çalışıldığından sonucu tahmin edilmek istenen her bir örnek için algoritma çıktıları olarak diyabet hastalığı yok ve diyabet hastalığı var anlamlarına gelen 0 ya da 1 değerleri döndürülmüştür. Algoritmaların döndürdükleri sonuçların 2^3 farklı şekilde gerçekleşebileceği söylenebilir. Tam sayımlama metodu ile gerçekleşebilecek bütün sonuçlar belirlenmiş ve Tablo 5.13'te gösterilmiştir.

Tablo 5.13. *Seçilen algoritmalara göre olabilecek tüm durumlar*

Naive Bayes	**Rastgele Ağaç (Karar Ağacı) RandomTree**	**Çok Katmanlı Algılayıcı (YSA)**
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Karma yöntemin oluşturulmasında izlenilen yol Şekil 5.10'da gösterilmiştir.



Şekil 5.10. *Hastalık teşhisinde izlenilen yol*

Veri setinde her bir örnek için seçilen üç yöntemden en az ikisinin tahmin sonucu aynıdır. Oylama yöntemi ile çoğunluğun tahmini karma yöntemin tahmin sonucu olarak kabul edilmiştir. Karma yöntemin tahmin sonuçlarına göre oluşturulan karmaşıklık matrisi Şekil 5.11'de verilmiştir.

		Tahmin Edilen Sınıf	
	n=37	0 (Diyabet Yok)	1 (Diyabet Var)
Gerçek Sınıf	0 (Diyabet Yok)	17	2
	1 (Diyabet Var)	0	18

Şekil 5.11. *Karma yöntemin karmaşıklık matrisi*

Karmaşıklık matrisine göre hesaplanan değerlendirme ölçütleri Tablo 5.14'te gösterilmiştir.

Tablo 5.14. *Karma yöntemin değerlendirme ölçütleri*

Değerlendirme Ölçütü	**Sonuç**
Doğru Sınıflandırma Oranı (%)	%94,585
Kesinlik (Precision)	0,9
Duyarlılık (Recall)	1
F Ölçütü (F measure)	0,947
Kappa İstatistiği	0,892

Karma yöntemin doğru sınıflandırma oranının ayrı ayrı hesaplanan doğru sınıflandırma oranlarından daha yüksek olduğu görülmüştür. Algoritmalar ayrı ayrı değerlendirildiğinde en yüksek doğru sınıflandırma oranı %91,892 iken karma yöntemin doğru sınıflandırma oranı %94,595 olarak hesaplanmıştır. Ayrı ayrı sınıflandırma yapıldığında en yüksek F ölçütü 0,923 iken karma yöntem için F ölçütü 0,947'ye yükselmiştir. Önceki durumda 0,838 olan en yüksek Kappa istatistiği değeri 0,892'ye

yükselmiştir. Pozitif tahmin edici değer olarak da bilinen kesinlik değeri diyabet hastalığı var olarak tahmin edilen test veri setindeki 20 kişinin gerçekte kaçının diyabet hastalığına sahip olduğunun ölçüsüdür. Hastalığa sahip olan 18 kişi doğru olarak tespit edildiğinden bu oran 0,90 olarak hesaplanmıştır. Duyarlılık ya da diğer bir deyişle gerçek pozitif oranı test veri setindeki gerçekte diyabet hastalığına sahip 18 kişiden kaç tanesinin doğru tahmin edildiğinin ölçüsüdür. Hastalığa sahip olduğu tahmin edilen kişi sayısı 18 olduğundan bu oran 1 olarak hesaplanmıştır. Yani diyabet hastalığına sahip olan kişiler doğru tespit edilmiştir. Ancak diyabet hastalığı olmayan 2 kişi için hastalık durumu "Hastalık var." olarak tahmin edilmiştir. Hastalığa sahip olmayan bir kişi için "Hastalık var." sonucunun alınması bu kişi için ek analizlerler yapılmasına neden olacaktır. Ama bu durum hastalığa sahip olan birisi için "Hastalık yok." sonucu alınmasından ve kişinin hastalığı ile detaylı bir tarama yapılmamasından daha kabul edilebilirdir. Kesinlik ve duyarlılık ölçütlerinin ödünleşmesi açısından bu ölçütlerin harmonik ortalaması alınarak hesaplanan F ölçütünün de yorumlanması daha doğru analiz yapılmasına yardımcı olmaktadır. 0,947 değeriyle en yüksek F ölçütü karma yöntemde elde edilmiştir.

Sınıflandırma yapılırken çoğu durumda tüm ana kütleye yani populasyona erişim mümkün olmamaktadır. Nadiren erişilmesinin mümkün olduğu durumlarda da hesaplamalar çok yüksek hesaplama zamanlarına ve maliyetlere sebep olmaktadır. Örneklemin anakütleyi ne kadarını temsil ettiğini ortaya çıkarmak gerekmektedir. Bu çalışmada örneklem ile sınıflandırma yapıldığından güven aralıklarının hesaplanmasına karar verilmiştir. Tıbbı çalışmalarda güven aralıklarının genellikle %95 olarak alındığı görülmüştür.

Karma yöntemin doğru sınıflandırma oranı, kesinlik ve duyarlılık değerlendirme ölçütleri için %95 güven seviyesinde güven aralığı (GA) hesaplanmıştır. GA hesaplanmasında MedCalc adlı istatistiksel bir yazılımdan yararlanılmıştır. Doğru sınıflandırma oranı ve duyarlılık için Clopper Pearson güven aralığı, kesinlik için Marceldo vd. (2007) çalışmasında kullandığı standart logit güven aralıkları kullanılmıştır.

Clopper Pearson güven aralıkları hesaplanmasında F dağılımı kullanılmaktadır. α anlamlılık düzeyini; a ve b serbestlik derecelerini ($F_{a,b\,;\alpha/2}$) göstermektedir. Duyarlılık d, doğru pozitif sayısı x, toplam pozitif sayısı n ile ifade edildiğinde ($\widehat{d} = \frac{x}{n}$) çift yönlü % 100 (1- α) güven aralığı (5.10)'daki gibidir [59]:

$$\left[\frac{\mathrm{x}}{\mathrm{x}+(\mathrm{n}-\mathrm{x}+1)F_{2(n-x+1),2x\,;\alpha/2}}\ ,\ \frac{(x+1)F_{2(x+1),2(n-x);\ \alpha/2}}{(n-x)+(x+1)F_{2(x+1),2(n-x);\ \alpha/2}}\right] \tag{5.10}$$

Standart logit güven aralıkları için kullanılan karmaşıklık matrisi Tablo 5.15'te verilmiştir.

Tablo 5.15. *Standart logit güven aralıkları için karmaşıklık matrisi*

	Hastalık-	**Hastalık+**
Tahmin-	x_{00}	x_{01}
Tahmin+	x_{10}	x_{11}
	n_0	n_1

Standart logit güven aralıkları hesaplanmasında kullanılan formüller aşağıdaki gibidir [60]:

$$\widehat{S_e} = \frac{x_{11}}{n_1} \tag{5.11}$$

$$\widehat{S_p} = \frac{x_{00}}{n_1} \tag{5.12}$$

$$p = \frac{x_{11}+x_{01}}{n_0+n_1} \tag{5.13}$$

Bu denklemlerde S_e duyarlılık değerini, S_p özgüllük değerini ve p prevelansı ifade etmektedir.

Pozitif prediktif değer olarak da bilinen kesinlik değeri için güven aralığı hesaplamasında kullanılan formüller şu şekildedir:

$$\text{Pozitif Prediktif Değer (PPV)} = \frac{S_e\,.\,p}{S_e\,.p+(1-S_p).(1-p)} \tag{5.14}$$

$$\text{logit (PPV)} = \log\left[\frac{S_e\,.\,p}{(1-S_p)(1-p)}\right] \tag{5.15}$$

$$\text{Var(PPV)}= \frac{[p\cdot(1-S_p)\cdot(1-p)]^2\,.\,S_e.\frac{1-S_e}{n_1} + [p.S_e.(1-p)]^2.\frac{S_p(1-S_p)}{n_0}}{\left[S_e.p+(1-S_p).(1-p)\right]^4} \tag{5.16}$$

$$\text{Var(logit(PPV))} = \left[\frac{1-S_e}{S_e}\right].\frac{1}{n_1} + \left[\frac{S_p}{1-S_p}\right].\frac{1}{n_0} \tag{5.17}$$

$$\text{Logit (PPV)Güven Aralığı}: \ \text{logit(PPV)} \pm z_{1-\alpha/2}\sqrt{\text{Var(logit(PPV))}} \qquad (5.18)$$

Güven aralığı orijinal haline geri dönüştürüldüğünde PPV için güven aralığı (5.18)'deki gibi hesaplanmaktadır.

$$\text{PPV}: \frac{e^{\text{logit (PPV)}-z_{1-\alpha/2}\sqrt{\text{Var(logit(PPV))}}}}{1+e^{\text{logit (PPV)}-z_{1-\alpha/2}\sqrt{\text{Var(logit(PPV))}}}} \ , \ \frac{e^{\text{logit (PPV)}+z_{1-\alpha/2}\sqrt{\text{Var(logit(PPV))}}}}{1+e^{\text{logit (PPV)}+z_{1-\alpha/2}\sqrt{\text{Var(logit(PPV))}}}} \qquad (5.19)$$

Karma yöntemin değerlendirme ölçütleri için hesaplanan güven aralıkları Tablo 5.16'da verilmiştir.

Tablo 5.16. *Karma yöntem için hesaplanan güven aralıkları*

Değerlendirme Ölçütü	**Değer**	**%95GA**
Doğru Sınıflandırma Oranı (%)	%94,585	%81,8-%99,34
Kesinlik (Precision)(%)	%90	%70,81-%97,09
Duyarlılık (Recall)	%100	%81,47-%100

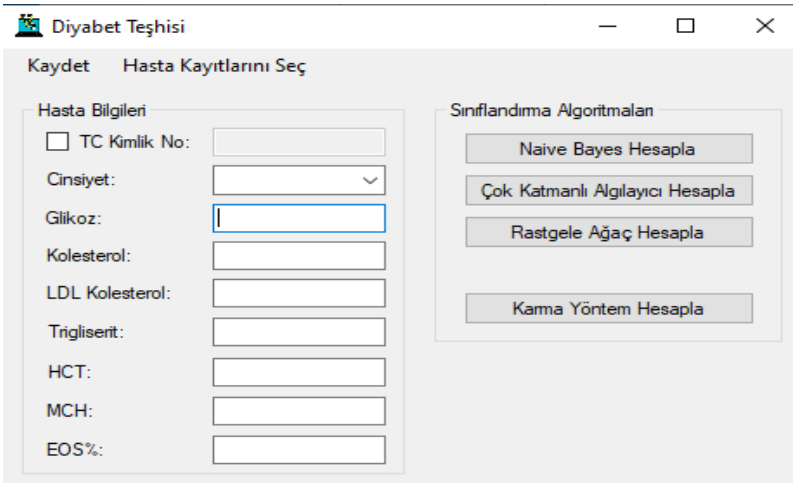
%95 güven seviyesinde (%5 anlamlılık düzeyinde) karma yöntemin anakütle üzerindeki doğru sınıflandırma oranının %81,8 ile %99,34 aralığında, kesinlik değerinin (%) %70,81 ile %97,09 aralığında ve duyarlılık değerinin (%) %81,47 ile %100 aralığında olduğu söylenebilir.

5.6. Diyabet Hastalığının Teşhisi için Karar Destek Sistemi Geliştirilmesi

Diyabet hastalığının tespit edilebilmesi için incelenen algoritmalardan en başarılı olan üç yöntem ve karma yöntem kullanılarak hastalara ait 8 adet giriş değerinin verilebileceği ve tahminlerin gerçekleştirilebileceği bir karar destek sistemi geliştirilmiştir. Visual Studio programında C# programlama dili kullanılarak tasarlanan karar destek sisteminin kullanıcı arayüzünde hasta bilgi girişinin yapılabildiği 8 adet metin kutusu bulunmaktadır. Arayüzde bulunan "Hasta Kayıtlarını Seç" butonu ile hastalara ait bilgilerin bulunduğu dosya seçilmektedir. Yeni bir veri girilmişse arayüzün sol üst köşesinde bulunan "Kaydet" butonu ile hasta bilgileri kaydedilmektedir. "Naive

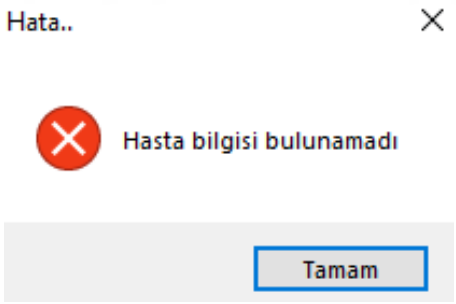
Bayes Hesapla", "Çok Katmanlı Algılayıcı Hesapla" ve "Rastgele Ağaç Hesapla" butonları ile sınıflandırma algoritmalarına göre bilgileri girilen kişilerin diyabet hastalığına sahip olma durumları değerlendirilmekte ve elde edilen sonuç ekrana bastırılmaktadır. Örneğin Naive Bayes algoritmasına göre bir sınıflandırma yapılmışsa ve tahminin sonucu kişinin diyabet hastalığına sahip olduğu yönündeyse ekranda "Naive Bayes algoritmasına göre kişi hastadır." bilgisinin olduğu mesaj kutusu görülmektedir. Bu algoritmaya göre kişinin diyabetli olmadığı tahmin edilmiş ise ekranda "Naive Bayes algoritmasına göre kişi hasta değildir." şeklinde bir yazı görülmektedir. "Karma Yöntem Hesapla" butonu ile algoritmalardan en az ikisinin tahmin sonucu karma yöntemin tahmin sonucunu oluşturmaktadır. Çoğunluğun tahmin oylamasına göre kişi hasta ise "Karma Yöntem Hesapla" butonuna basıldığında "Karma yöntemin tahmin sonucuna göre kişi hastadır." yazısı ekrana bastırılmaktadır. Çoğunluğun tahmin oylamasına göre kişi hasta değil ise "Karma yöntemin tahmin sonucuna göre kişi hasta değildir." yazısı mesaj kutusu ile ekrana bastırılmaktadır.

Oluşturulan arayüz, değerlendirici ve karar vericinin algoritmaların alt yapısını, hesapsal boyutunu ve karmaşık işlemleri görmeden sadece hasta bilgileri girerek sonuçlara ulaşabilmesini sağlamaktadır. WEKA'da sınıflandırma yapılırken eğitim ve test verileri programa aktarılabilse de model başarısı test edildikten sonra yeni gelen bir veriye göre sınıflandırma algoritmalarının tahmin sonuçları ayrı ayrı sınıflandırma işlemi yapılması sonucunda elde edilebilmektedir. Bu çalışma kapsamında oluşturulan karma modelin tahmin sonucu tek bir buton ile karar vericiye aktarılmaktadır. Oluşturulan kullanıcı arayüzüne ait görsel Şekil 5.12'de verilmiştir.



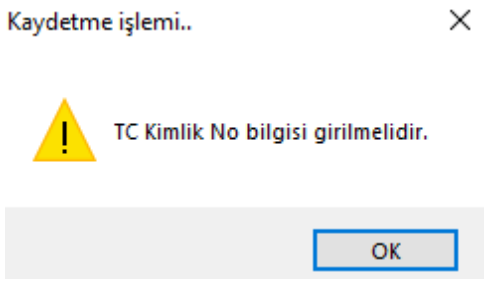
Şekil 5.12. *Diyabet teşhisi için kullanıcı arayüzü*

Bilgi girişi yapılan örnek sistemde kayıtlı değilse "Hasta bilgisi bulunamadı." Şeklinde bir uyarı mesajı ile karşılaşılmaktadır. Uyarı ekranı Şekil 5.13'teki gibidir. Bu durumda yeni veri girişi yapıldıktan sonra örneğin kaydet butonu ile kaydedilmesi gerekmektedir.



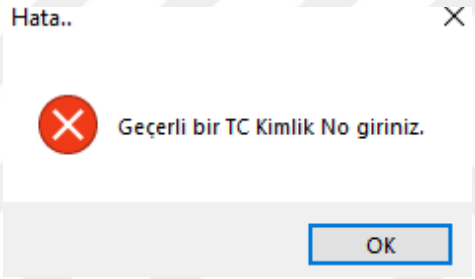
Şekil 5.13. *Hasta bilgisi bulunamadığında verilen uyarı ekranı*

Hasta kaydı yapılırken tahlil sonuçlarına ek olarak TC Kimlik No bilgisinin girilmesi gerekmektedir. TC Kimlik No bilgisinin girilmediği durumda Şekil 5.14'te gösterilen "TC Kimlik No bilgisi girilmelidir." şeklinde bir mesaj kutusu ile karşılaşılmaktadır.



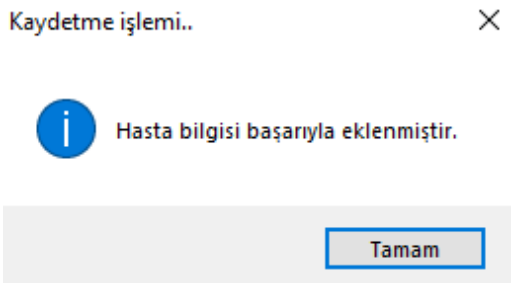
Şekil 5.14. *TC Kimlik No bilgisi girilmediğinde verilen uyarı mesajı*

Yanlış veya hatalı bir TC Kimlik No girildiği durumda kaydetme işleminde Şekil 5.15'te gösterilen "Geçerli bir TC Kimlik No giriniz." uyarı mesajı ile karşılaşılmaktadır.



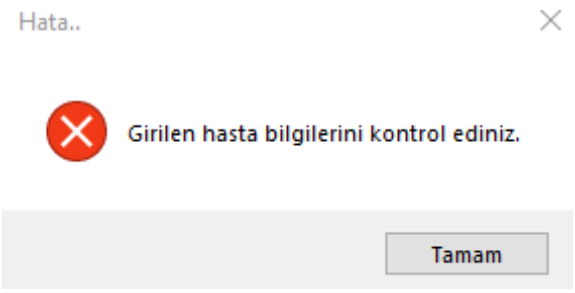
Şekil 5.15. *Yanlış veya hatalı TC Kimlik No girildiğinde verilen uyarı mesajı*

Arayüzün sol üst köşesindeki "Kaydet" butonu ile yeni hasta kaydı yapıldıktan sonra yeni kaydın başarı ile yapıldığını gösteren "Hasta bilgileri başarıyla eklenmiştir." şeklinde bir bilgi mesajı verilmektedir. Bilgi mesajına ait görsel Şekil 5.16'da verilmiştir.



Şekil 5.16. *Kaydetme işlemi yapılığında verilen bilgi mesajı*

Ara yüze yanlış formatta bilgi girişi yapılmışsa “Girilen hasta bilgilerini kontrol ediniz.” şeklinde bir hata mesajı ile karşılaşılmaktadır. Yanlış bilgi girişi yapıldığında verilen hata mesajına ait görsel şekil 5.17’de verilmiştir.



Şekil 5.17. *Yanlış bilgi girişi yapıldığında verilen hata mesajı*

6. SONUÇ VE ÖNERİLER

Günümüzde sağlık alanında artan veri miktarı ile birlikte insan gözünün yanılabileceği de düşünüldüğünde karar verme sürecini destekleyecek yeni araçların gerekliliği veri madenciliği tekniklerinin bu alandaki kullanımını yaygın hale getirmiştir.

Bu çalışmada diyabet şüphesi taşıyan ve Adana'da hizmet veren bir sağlık kuruluşuna başvuran hastalara ait kan tahlili değerleri kullanılarak diyabet hastalığının sınıflandırma algoritmalarının birlikte kullanılmasına dayalı bir yöntem ile teşhis edilmesi üzerinde çalışılmıştır. Hastalığa sahip tüm insanların olduğu anakütleye erişim mümkün olmadığından kullanılan değerlendirme ölçütleri için güven aralıkları hesaplanmıştır. Oluşturulan karma yöntemin kullanılabilir olması için Visual Studio'da C# programlama dili ile kullanıcının karar vermesini kolaylaştırıcı bir kullanıcı arayüzü tasarlanmıştır.

Literatürde diyabet hastalığının sınıflandırılması ile ilgili çalışmalar incelendiğinde çalışmaların çoğunun sınırlı sayıda özniteliğe sahip, eski bir veri seti üzerinde yoğunlaştığı ve çalışmalarda genellikle en doğru sınıflandırmayı yapan yöntemin tespit edilmesi üzerine odaklanıldığı ya da yöntemler arasında karşılaştırmalar yapıldığı görülmüştür. Bu veri setinin eski olduğu ve zamanla hastalık üzerinde etkili olabilecek farklı değişkenlerin de ortaya çıkabileceği düşüncesiyle bu çalışmada yeni bir veri seti ile çalışılması amaçlanmış ve dosya taraması yoluyla hastalara ait bilgiler toplanmıştır. Ayrıca literatürdeki çalışmalardan farklı olarak öznitelik seçiminde PROC VARCLUS yöntemi kullanılmıştır. Bu yöntemde değişkenler kümelenmekte ve kümedeki değişkenliği en çok açıklayanlar küme temsilcileri olarak seçilmektedir. Seçilen öznitelikler sınıflandırma algoritmalarında kullanılabilmektedir.

WEKA kütüphanesinde bulunan ve çalışma kapsamında incelenen en başarılı üç sınıflandırma yöntemi olan Naive Bayes, rastgele ağaç karar ağacı ve çok katmanlı algılayıcı (yapay sinir ağı) sınıflandırıcılarının çoğunluğun kararına göre birlikte değerlendirilmesine dayalı olarak oluşturulan karma yöntem ile sınıflandırma yapılmıştır. Algoritmaların bireysel değerlendirilmesi sonucu en yüksek doğru sınıflandırma oranı %91,892 iken karma yöntemin doğru sınıflandırma oranı %94,595 olarak bulunmuştur. Bireysel değerlendirmelerde en yüksek F ölçütü 0,923 iken karma yöntem için F ölçütü 0,947'ye yükselmiştir. Önceki durumda 0,838 olan en yüksek Kappa istatistiği değeri 0,892'ye, kesinlik değeri 0,895'ten 0,9'a yükselmiştir. Karma yöntemin anakütleyi ne

kadar temsil ettiğinin belirlenebilmesi için oluşturulan güven aralıklarına göre %95 güven seviyesinde (%5 anlam düzeyinde) karma yöntemin anakütle üzerindeki doğru sınıflandırma oranının %81,8 ile %99,34 aralığında, kesinlik değerinin (%) %70,81 ile %97,09 aralığında ve duyarlılık değerinin (%) %81,47 ile %100 aralığında olduğu söylenebilir.

Karma yöntemin gerçek hayatta da kullanılabilir olması için C# ile tasarlanan arayüz vasıtasıyla hasta kayıtlarının sisteme girilebildiği, hasta kayıt dosyasının seçilebildiği, kullanılan yöntemler ve karma yöntemin sonuçlarının alınabildiği, veri madenciliği ve sınıflandırma işlemleri açısından çok fazla bilgi ve uzmanlık gerektirmeyen, kullanıcının karar vermesini kolaylaştırıcı bir yapı oluşturulmuştur.

Sonraki çalışmalarda bu arayüzü kullanacak hekimler için verilerin ortak bir veritabanında toplanacağı, kullanım ile ilgili şartlar ve gizlilik sözleşmesi bilgilerinin bulunduğu bir web arayüzü ve mobil uygulama tasarlanabilir. Böylelikle daha geniş kitlelerin katılım sağladığı bir veritabanı ile daha faydalı sonuçlara ulaşılabilir. Önerilen yöntem, farklılaştırılmış öznitelikler ile sonraki çalışmalarda farklı hastalıkların teşhisinde de kullanılabilir.

KAYNAKÇA

[1] https://www.idf.org/aboutdiabetes/what-is-diabetes.html

(Erişim Tarihi:05.05.2020)

[2] https://www.who.int/health-topics/diabetes#tab=tab_1

(Erişim Tarihi:06.05.2020)

[3] International Diabetes Federation. (2019). *IDF Diabetes Atlas*. (9th edition). Brussels: International Diabetes Federation.

[4] Witten, I.H., Frank, E., Hall, M.A., Pal, C.J. (2017). *Data mining: practical machine learning tools and techniques*. (4th Edition). Cambridge: Morgan Kaufmann

[5] Phyu, T.N. (2009). Survey of classification techniques in data mining. *International Multi Conference of Engineers and Computer Scientists*, 28-20.

[6] Das, H., Naik, B., Behera, H.S. (2018). Classification of diabetes mellitus disease (DMD): A data mining (DM) approach.

[7] Kannadasan, K., Edla, D.R. and Kuppili, V. (2019). Type 2 diabetes data classification using stacked autoencoders in deep neural networks. *Clinical Epidemiology and Global Health*, 7, 530–535.

[8] Ismi, D.P., Panchoo, S. and Murinto, M. (2016). K-means clustering based filter feature selection on high dimensional data. *International Journal of Advances in Intelligent Informatics*, 2(1), 38-45.

[9] Patricio, M., Pereira, J., Crisóstomo,j., Matafome, P.,Gomes, M., Seica, R.Caramelo, F. (2018). Using Resistin, glucose, age and BMI to predict the presence of breast cancer. *BMC Cancer*, 18, 21-29.

[10] Sobar, R. M. and Wijaya, A. (2016). Behavior determinant based cervical cancer early detection with machine learning algorithm. *Advanced Science Letters*, 22(10), 3120-3123.

[11] Mumtaz, W., Xia, L., Ali, S. S. A., Yasin, M. A. M., Hussain, M., Malik, A. S. (2017). Electroencephalogram (EEG)-based computer-aided technique to diagnose major depressive disorder (MDD). *Biomedical Signal Processing and Control*, 31, 108-115.

[12] Azam, M. S., Raihan, M. A. and Rana, H. K. (2020). An experimental study of various machine learning approaches in heart disease prediction. *International Journal of Computer Applications*, 175(21), 16-21.

[13] Alassaf, R. A., Alsulaim, K. A., Alroomi, N. Y., Alsharif, N. S., Aljubeir, M. F., Olatunji, S. O., Alahmadi, A. Y., Imran, M., Alzahrani, R. A., Alturayeif, N S. (2018). Preemptive Diagnosis of Diabetes Mellitus Using Machine Learning. *Saudi Computer Society National Computer Conference (NCC)*, Riyadh.

[14] Nimmala, S. (2019). Machine Learning approaches to classify diabetes patients based on age, obesity level and cholesterol level. *CVR Journal of Science and Technology*, 16, 97-101.

[15] Daghistani, T. and Alshammari, R. (2020). Comparison of statistical logistic regression and random forest machine learning techniques in predicting diabetes. *Journal of Advances in Information Technology*, 11(2), 78-83.

[16] Selvakumar, S., Kannan, K.S. and GothaiNachiyar, S. (2017). Prediction of diabetes diagnosis using classification based data mining techniques. *International Journal of Statistics and Systems*, 12(2), 183-188.

[17] Iyer, A., Jeyalatha, S., Sumbaly, R. (2015). Diagnosis of diabetes using classification mining techniques. *International Journal of Data Mining & Knowledge Management Process (IJDKP)*, 5, 1-14.

[18] Kurt, M. S. ve Ensari, T. (2017). Diabet diagnosis with support vector machines and multi layer perceptron. *Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT)*, 1-4.

[19] Kumari, V.A., Chitra, R. (2013). Classification of diabetes disease using support vector machine. *International Journal of Engineering Research and Applications*, 3, 1797-1801.

[20] Kaur, G., Chhabra, A. (2014). Improved J48 classification algorithm for the prediction of diabetes. *International Journal of Computer Applications*, 98(22), 13-17.

[21] Bozkurt, M.R., Yurtay, N., Yılmaz, Z., Sertkaya, C. (2014). Comparison of different methods for determining diabetes. *Turkish Journal of Electrical Engineering & Computer Sciences*, 22, 1044-1055.

[22] Garg, S. B., Mahajan, A. K. and Kamal, T.S. (2017). An approach for diabetes detection using data mining classification techniques. *International Journal of Engineering Sciences*, 26, 202-218.

[23] Radha, P. and Srinivasan, B. (2014). Predicting diabetes by cosequencing the various data mining classification techniques. *International Journal of Innovative Science, Engineering & Technology*, 1(6).

[24] Akyol, K. and Şen, B. (2018). Diabetes mellitus data classification by cascading of feature selection methods and ensemble learning algorithms. *I.J. Modern Education and Computer Science*, 6, 10-16.

[25] Kumar, S. P. and Umatejaswi, V. (2017). Diagnosing diabetes using data mining techniques. *International Journal of Scientific and Research Publications*, 7(6), 705-709.

[26] Bashir, S., Qamar, U., Khan, F.H., Javed, M.Y. (2014). An efficient rule-based classification of diabetes Using ID3, C4.5 & CART ensembles. *12th International Conference on Frontiers of Information Technology*, 226-231.

[27] Zhu, J., Xie, Q. and Zheng, K. (2015). An improved early detection method of type-2 diabetes mellitus using multiple classifier system. *Information Sciences*, 292, 1–14.

[28] Dewangan, A. K. and Agrawal, P. (2015). Classification of diabetes mellitus using machine learning techniques. *International Journal of Engineering and Applied Sciences (IJEAS)*, 2(5), 145-148.

[29] Devi, R. D. H., Bai, A. and Nagarajan, N. (2020). A novel hybrid approach for diagnosing diabetes mellitus using farthest first and support vector machine algorithms. *Obesity Medicine*, 17.

[30] Vinupritha, P., Hariharan, M., Kathirvelu, D., Chinnadurai, S. (2017). Estimation of hemoglobin A1c using the complete blood count measures in the diagnosis of diabetes. *Asian Journal of Pharmaceutical and Clinical Research*, 10(9), 214-218.

[31] Shehri, Z. S. A. (2017). The relationship between some biochemical and hematological changes in type 2 diabetes mellitus. *Biomedical Research & Therapy*, 4(11), 1760-1774.

[32] Giudici, P. (2003). *Applied data mining: Statistical methods for business and industry*. New York: J. Wiley.

[33] Han, J., Kamber, M. and Pei, J. (2012). *Data mining: Concepts and techniques.* (3rd Edition). Waltham: Morgan Kaufmann.

[34] Tan, P. N., Steinbach, M. and Kumar,V. (2006). *Introduction to data mining*. USA: Addison-Wesley.

[35] Taşçı, E. ve Onan, A. (2016). K en yakın komşu algoritması parametrelerinin sınıflandırma performansı üzerine etkisinin incelenmesi, *Akademik Bilişim*, 1-8.

[36] Öztemel, E. (2012). *Yapay sinir ağları.* (3.baskı). İstanbul: Papatya Yayıncılık.

[37] Çayıroğlu İ. (2015). *Yapay sinir ağları.* Karabük: Karabük Üniversitesi.

[38] Akşehirli, Ö., Ankaralı, H., Aydın, D., Saraçlı, Ö. (2013). Tıbbi tahminde alternatif bir yaklaşım: Destek vektör makineleri. *Turkiye Klinikleri Journal of Biostatistics*, 5(1), 19-28.

[39] Ayhan, S. ve Erdoğmuş, Ş. (2014). Destek vektör makineleriyle sınıflandırma problemlerinin çözümü için çekirdek fonksiyonu seçimi. *Eskişehir Osmangazi Üniversitesi İBBF Dergisi*, 9(1), 175-198.

[40] Onan, A. (2015). Şirket iflaslarının tahmin edilmesinde karar ağacı algoritmalarının karşılaştırmalı başarım analizi. *Bilişim Teknolojileri Dergisi*, 8(1), 9-19.

[41] Kiranmai, S.A. and Laxmi, A. J. (2018). Data mining for classification of power quality problems using WEKA and the effect of attributes on classification accuracy. *Protection and Control of Modern Power Systems*, 3(3),303-314.

[42] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5-32.

[43] Sanche, R. and Lonergan, K. (2006). Variable reduction for predictive modeling with clustering. *Casualty Actuarial Society Forum*, 89-100.

[44] Pasta, D. and Suhr, D. (2004). Creating scales from questionnaires: PROC VARCLUS vs. Factor Analysis. *SAS Users Group International Conference*, Montreal, Canada: SAS Institute.

[45] Johnson, R.A. and Wichern, D.W. (2007). *Applied Multivariate Statistical Analysis*. (6th edition). New Jersey: Prentice-Hall.

[46] Fabriger, L.R., MacCallum, R.C., Wegener, D.T., Strahan, E.J. (1999). Evaluating the use of exploratory factor analysis in psychological research. *Psychological Methods*, 4, 272-299.

[47] Velicer, W.F. and Jackson, D.N. (1990). Component analysis versus common factor analysis: Some issue in selecting and appropriate procedure. *Multivariate Behavioral Research*, 25(1), 1-28.

[48] SAS Institute Inc. (2017). *SAS/STAT® 14.3 User's Guide*. Cary, NC: SAS Institute Inc

[49] SAS Institute Inc. (2018). *JMP® 14 Multivariate Methods*. Cary, NC: SAS Institute Inc.

[50] Aggarwal, V. and Kosian, S. (2011). Feature selection and dimension reduction techniques in SAS, NESUG.

[51] https://www.turkdiab.org/diyabet-hakkinda-hersey.asp?lang=TR&id=46

(Erişim Tarihi:20.11.2020)

[52] https://www.idf.org/our-activities/care-prevention/gdm

(Erişim Tarihi:20.11.2020)

[53] http://www.diabetcemiyeti.org/c/farkli-bir-diyabet-tipi-mody-tip-genclerde-gorulen-eriskin-tipi-diyabet

(Erişim Tarihi:20.11.2020)

[54] https://www.idf.org/aboutdiabetes/complications.html

(Erişim Tarihi:20.11.2020)

[55] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H. (2009). The WEKA data mining software: An update. *SIGKDD*, 11, 10-18.

[56] Powers, D.M.W. (2011). Evaluation: From precision recall and f-measure to roc informedness markedness and correlation. *Journal of Machine Learning Technologies*, 2(1), 37-63.

[57] Aksu, G. (2018). *PISA başarısını tahmin etmede kullanılan veri madenciliği yöntemlerinin incelenmesi.* Yayımlanmamış Doktora Tezi. Ankara: Hacettepe Üniversitesi, Eğitim Bilimleri Enstitüsü.

[58] Landis, J.R. and Koch, G.G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33, 150-174.

[59] Fleiss, J.L., Levin, B. and Paik, M.C. (2003). *Statistical Methods for Rates and Proportions.* (3rd Edition). New Jersey: John Wiley & Sons.

[60] Mercaldo, N.D., Lau, K.F. and Z, X.H. (2007). Confidence intervals for predictive values with an emphasis to case–control studies. *Statistics In Medicine*, 26(10), 2170-2183.

EKLER

EK-1 Visual Studio- C# Kodları

EK-2 Eskişehir Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü 87914409-050.99 Sayılı ve 20.11.2020 Tarihli Etik Kurul Karar Belgesi

EK-1 Visual Studio- C# Kodları

Hasta Bilgileri

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DiyabetArayüz
{
   class HastaBilgileri
   {
      public  enum   CinsiyetEnum : int
      {
      Kadın = 0,
      Erkek = 1
      }

       public static UInt64 TCNo;
       public static int cinsiyet = 0;
       public static double trigliserit;
       public static double glikoz;
       public static double ldlkolesterol;
       public static double kolesterol;
       public static double hct;
       public static double mch;
       public static double eos;

       public static int  standardizeCinsiyet = 0;
       public static double standardizeTrigliserit;
       public static double standardizeGlikoz;
       public static double standardizeLdlkolesterol;
       public static double standardizeKolesterol;
       public static double standardizeHct;
       public static double standardizeMch;
       public static double standardizeEos;

       public static double ortalamaTrigliserit;
       public static double ortalamaGlikoz;
       public static double ortalamaLdlkolesterol;
       public static double ortalamaKolesterol;
       public static double ortalamaHct;
       public static double ortalamaMch;
       public static double ortalamaEos;

       public static double standartSapmaTrigliserit;
       public static double standartSapmaGlikoz;
       public static double standartSapmaLdlkolesterol;
```

```
        public static double standartSapmaKolesterol;
        public static double standartSapmaHct;
        public static double standartSapmaMch;
        public static double standartSapmaEos;

        public static int mevcutHastaNumarası = 0;
        public static bool TCNo_Aktif = false;
    }
}
```

Excel Fonksiyonları

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Excel = Microsoft.Office.Interop.Excel;

namespace DiyabetArayüz
{
    class ExcelFonksiyonları
    {
        public static bool hastaBilgileriMevcutMu(Excel.Worksheet worksheet, int
satırSayısı)
        {
            try
            {
                for (int i = 2; i < satırSayısı+1; i++)
                {
                    if (
                    (int)(worksheet.Cells[i, 1] as Excel.Range).Value == HastaBilgileri.cinsiyet
&&
                    (double)(worksheet.Cells[i, 2] as Excel.Range).Value ==
HastaBilgileri.glikoz &&
                    (double)(worksheet.Cells[i, 3] as Excel.Range).Value ==
HastaBilgileri.kolesterol &&
                    (double)(worksheet.Cells[i, 4] as Excel.Range).Value ==
HastaBilgileri.ldlkolesterol &&
                    (double)(worksheet.Cells[i, 5] as Excel.Range).Value ==
HastaBilgileri.trigliserit &&
                    (double)(worksheet.Cells[i, 6] as Excel.Range).Value == HastaBilgileri.hct
&&
                    (double)(worksheet.Cells[i, 7] as Excel.Range).Value == HastaBilgileri.mch
&&
```

```
                (double)(worksheet.Cells[i, 8] as Excel.Range).Value == HastaBilgileri.eos
                    )
                {
                    HastaBilgileri.mevcutHastaNumarası = i;
                    return true;
                }
            }
        }
        catch
        {
            return false;
        }

        return false;
    }

    public static void CloseExcels()
    {
        foreach (var process in Process.GetProcessesByName("EXCEL"))
        {
            try
            {
                process.Kill();
            }
            catch
            {

            }
        }
    }

    public static void standardizeKayıtEkle(Excel.Worksheet
worksheet_EğitimVeriSeti, Excel.Worksheet worksheet_StandardizeYeniKayıt, int
yeniSatırNumarası_YeniKayıt)
    {
        worksheet_EğitimVeriSeti.Cells[1, 101] = "=STDEV.S(B:B)";
        worksheet_EğitimVeriSeti.Cells[1, 102] = "=STDEV.S(C:C)";
        worksheet_EğitimVeriSeti.Cells[1, 103] = "=STDEV.S(D:D)";
        worksheet_EğitimVeriSeti.Cells[1, 104] = "=STDEV.S(E:E)";
        worksheet_EğitimVeriSeti.Cells[1, 105] = "=STDEV.S(F:F)";
        worksheet_EğitimVeriSeti.Cells[1, 106] = "=STDEV.S(G:G)";
        worksheet_EğitimVeriSeti.Cells[1, 107] = "=STDEV.S(H:H)";

        worksheet_EğitimVeriSeti.Cells[2, 101] = "=AVERAGE(B:B)";
        worksheet_EğitimVeriSeti.Cells[2, 102] = "=AVERAGE(C:C)";
        worksheet_EğitimVeriSeti.Cells[2, 103] = "=AVERAGE(D:D)";
        worksheet_EğitimVeriSeti.Cells[2, 104] = "=AVERAGE(E:E)";
        worksheet_EğitimVeriSeti.Cells[2, 105] = "=AVERAGE(F:F)";
        worksheet_EğitimVeriSeti.Cells[2, 106] = "=AVERAGE(G:G)";
```

```
            worksheet_EğitimVeriSeti.Cells[2, 107] = "=AVERAGE(H:H)";

            HastaBilgileri.standartSapmaGlikoz =
(double)(worksheet_EğitimVeriSeti.Cells[1, 101] as Excel.Range).Value;
            HastaBilgileri.standartSapmaKolesterol =
(double)(worksheet_EğitimVeriSeti.Cells[1, 102] as Excel.Range).Value;
            HastaBilgileri.standartSapmaLdlkolesterol =
(double)(worksheet_EğitimVeriSeti.Cells[1, 103] as Excel.Range).Value;
            HastaBilgileri.standartSapmaTrigliserit =
(double)(worksheet_EğitimVeriSeti.Cells[1, 104] as Excel.Range).Value;
            HastaBilgileri.standartSapmaHct = (double)(worksheet_EğitimVeriSeti.Cells[1,
105] as Excel.Range).Value;
            HastaBilgileri.standartSapmaMch = (double)(worksheet_EğitimVeriSeti.Cells[1,
106] as Excel.Range).Value;
            HastaBilgileri.standartSapmaEos = (double)(worksheet_EğitimVeriSeti.Cells[1,
107] as Excel.Range).Value;

            HastaBilgileri.ortalamaGlikoz = (double)(worksheet_EğitimVeriSeti.Cells[2,
101] as Excel.Range).Value;
            HastaBilgileri.ortalamaKolesterol = (double)(worksheet_EğitimVeriSeti.Cells[2,
102] as Excel.Range).Value;
            HastaBilgileri.ortalamaLdlkolesterol =
(double)(worksheet_EğitimVeriSeti.Cells[2, 103] as Excel.Range).Value;
            HastaBilgileri.ortalamaTrigliserit = (double)(worksheet_EğitimVeriSeti.Cells[2,
104] as Excel.Range).Value;
            HastaBilgileri.ortalamaHct = (double)(worksheet_EğitimVeriSeti.Cells[2, 105]
as Excel.Range).Value;
            HastaBilgileri.ortalamaMch = (double)(worksheet_EğitimVeriSeti.Cells[2, 106]
as Excel.Range).Value;
            HastaBilgileri.ortalamaEos = (double)(worksheet_EğitimVeriSeti.Cells[2, 107]
as Excel.Range).Value;

            HastaBilgileri.standardizeGlikoz = (HastaBilgileri.glikoz -
HastaBilgileri.ortalamaGlikoz) / HastaBilgileri.standartSapmaGlikoz;
            HastaBilgileri.standardizeKolesterol = (HastaBilgileri.kolesterol -
HastaBilgileri.ortalamaKolesterol) / HastaBilgileri.standartSapmaKolesterol;
            HastaBilgileri.standardizeLdlkolesterol = (HastaBilgileri.ldlkolesterol -
HastaBilgileri.ortalamaLdlkolesterol) / HastaBilgileri.standartSapmaLdlkolesterol;
            HastaBilgileri.standardizeTrigliserit    = (HastaBilgileri.trigliserit   -
HastaBilgileri.ortalamaTrigliserit       ) /  HastaBilgileri.standartSapmaTrigliserit      ;
            HastaBilgileri.standardizeHct          = (HastaBilgileri.hct          -
HastaBilgileri.ortalamaHct          ) /  HastaBilgileri.standartSapmaHct         ;
            HastaBilgileri.standardizeMch          = (HastaBilgileri.mch          -
HastaBilgileri.ortalamaMch          ) /  HastaBilgileri.standartSapmaMch         ;
            HastaBilgileri.standardizeEos          = (HastaBilgileri.eos          -
HastaBilgileri.ortalamaEos          ) / HastaBilgileri.standartSapmaEos;
```

```
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 1] =
HastaBilgileri.cinsiyet;
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 2] =
HastaBilgileri.standardizeGlikoz;
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 3] =
HastaBilgileri.standardizeKolesterol;
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 4] =
HastaBilgileri.standardizeLdlkolesterol;
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 5] =
HastaBilgileri.standardizeTrigliserit;
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 6] =
HastaBilgileri.standardizeHct;
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 7] =
HastaBilgileri.standardizeMch;
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 8] =
HastaBilgileri.standardizeEos;
            worksheet_StandardizeYeniKayıt.Cells[yeniSatırNumarası_YeniKayıt, 9] =
HastaBilgileri.TCNo;
        }

        public static void standardizeKayıtOku(Excel.Worksheet
worksheet_StandardizeDiyabetVeriSeti)
        {
            HastaBilgileri.standardizeCinsiyet =
(int)(worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumaras
ı, 1] as Excel.Range).Value;
            HastaBilgileri.standardizeGlikoz =
(double)(worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNum
arası, 2] as Excel.Range).Value;
            HastaBilgileri.standardizeKolesterol =
(double)(worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNum
arası, 3] as Excel.Range).Value;
            HastaBilgileri.standardizeLdlkolesterol =
(double)(worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNum
arası, 4] as Excel.Range).Value;
            HastaBilgileri.standardizeTrigliserit =
(double)(worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNum
arası, 5] as Excel.Range).Value;
            HastaBilgileri.standardizeHct =
(double)(worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNum
arası, 6] as Excel.Range).Value;
            HastaBilgileri.standardizeMch =
(double)(worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNum
arası, 7] as Excel.Range).Value;
            HastaBilgileri.standardizeEos =
(double)(worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNum
arası, 8] as Excel.Range).Value;
        }
```

```
        public static void yeniHastaEkle(Excel.Worksheet worksheet, int
yeniSatırNumarası)
        {
            worksheet.Cells[yeniSatırNumarası, 1] = HastaBilgileri.cinsiyet;
            worksheet.Cells[yeniSatırNumarası, 2] = HastaBilgileri.glikoz;
            worksheet.Cells[yeniSatırNumarası, 3] = HastaBilgileri.kolesterol;
            worksheet.Cells[yeniSatırNumarası, 4] = HastaBilgileri.ldlkolesterol;
            worksheet.Cells[yeniSatırNumarası, 5] = HastaBilgileri.trigliserit;
            worksheet.Cells[yeniSatırNumarası, 6] = HastaBilgileri.hct;
            worksheet.Cells[yeniSatırNumarası, 7] = HastaBilgileri.mch;
            worksheet.Cells[yeniSatırNumarası, 8] = HastaBilgileri.eos;
            worksheet.Cells[yeniSatırNumarası, 9] = HastaBilgileri.TCNo;
        }

        public static void excelKaydetmedenKapat(Excel.Workbook workbook,
Excel.Application excel, Excel.Range userRange, Excel.Worksheet worksheet1,
Excel.Worksheet worksheet2 = null, Excel.Worksheet worksheet3 = null)
        {
            workbook.Close(0);
            excel.Quit();
            if (userRange != null) Marshal.ReleaseComObject(userRange);
            if (workbook != null) Marshal.ReleaseComObject(workbook);
            if (worksheet1 != null) Marshal.ReleaseComObject(worksheet1);
            if (worksheet2 != null) Marshal.ReleaseComObject(worksheet1);
            if (worksheet3 != null) Marshal.ReleaseComObject(worksheet1);
            if (excel != null) Marshal.ReleaseComObject(excel);
            ExcelFonksiyonları.CloseExcels();
        }

        public static void excelKaydetVeKapat(Excel.Workbook workbook,
Excel.Application excel, Excel.Range userRange, Excel.Worksheet worksheet1,
Excel.Worksheet worksheet2 = null, Excel.Worksheet worksheet3 = null,
Excel.Worksheet worksheet4 = null, Excel.Worksheet worksheet5 = null)
        {
            string myPath = Properties.Settings.Default.HastaKayıtları;
            string myPathTemp =
Path.GetDirectoryName(Path.GetDirectoryName(System.IO.Directory.GetCurrentDirec
tory())) + "HastaKayıtlarıTemp.xlsx";

            workbook.Close(true, myPathTemp);
            excel.Quit();
            if (userRange != null) Marshal.ReleaseComObject(userRange);
            if (workbook != null) Marshal.ReleaseComObject(workbook);
            if (worksheet1 != null) Marshal.ReleaseComObject(worksheet1);
            if (worksheet2 != null) Marshal.ReleaseComObject(worksheet2);
            if (worksheet3 != null) Marshal.ReleaseComObject(worksheet3);
            if (worksheet4 != null) Marshal.ReleaseComObject(worksheet4);
            if (excel != null) Marshal.ReleaseComObject(excel);
            ExcelFonksiyonları.CloseExcels();
```

```
            try
            {
                System.Threading.Thread.Sleep(1000);
                File.Delete(myPath);
                File.Move(myPathTemp, myPath);
            }
            catch
            {
                System.Threading.Thread.Sleep(2000);
                File.Delete(myPath);
                File.Move(myPathTemp, myPath);
            }
        }
    }
}
```

Sınıflandırma Algoritmaları

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DiyabetArayüz
{
    class SınıflandırmaAlgoritmaları
    {
        public static bool rastgeleAğaçHesapla()
        {

            if (HastaBilgileri.standardizeTrigliserit < -0.19)
            {
                if (HastaBilgileri.standardizeGlikoz < -0.27)
                {
                    if (HastaBilgileri.standardizeMch < -0.89)
                    {
                        if (HastaBilgileri.standardizeKolesterol < -0.71)
                            return true;
                        else if (HastaBilgileri.standardizeKolesterol >= -0.71)
                            return false;
                    }
                    else if (HastaBilgileri.standardizeMch >= -0.89)
                        return false;
                }
                else if (HastaBilgileri.standardizeGlikoz >= -0.27)
                {
                    if (HastaBilgileri.standardizeLdlkolesterol < -0.89)
                    {
```

```
                if (HastaBilgileri.standardizeGlikoz < 0.05)
                    return false;
                else if (HastaBilgileri.standardizeGlikoz >= 0.05)
                    return true;
            }
            else if (HastaBilgileri.standardizeLdlkolesterol >= -0.89)
                return true;
        }

    }
    else if (HastaBilgileri.standardizeTrigliserit >= -0.19)
    {
    if (HastaBilgileri.standardizeMch < 0.24)
        {
            if (HastaBilgileri.standardizeLdlkolesterol < 1.22)
                return true;
            else if (HastaBilgileri.standardizeLdlkolesterol >= 1.22)
                return false;
        }
    else if (HastaBilgileri.standardizeMch >= 0.24)
         {
            if (HastaBilgileri.standardizeGlikoz < -0.53)
                return false;
            else if (HastaBilgileri.standardizeGlikoz >= -0.53)
                return true;
        }
    }
    return false;
}

    public static bool
ÇokKatmanlıAlgılayıcı(Microsoft.Office.Interop.Excel.Worksheet
worksheet_ÇokKatmanlıAlgılayıcı, Microsoft.Office.Interop.Excel.Worksheet
worksheet_StandardizeDiyabetVeriSeti)
    {
        worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 1] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 1];
        worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 2] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 2];
        worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 3] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 3];
        worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 4] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 4];
        worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 5] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 5];
        worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 6] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 6];
        worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 7] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 7];
```

```
            worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 8] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 8];

            double Node0Sonuç = (double)(worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 16]
as Microsoft.Office.Interop.Excel.Range).Value;
            double Node1Sonuç = (double)(worksheet_ÇokKatmanlıAlgılayıcı.Cells[2, 17]
as Microsoft.Office.Interop.Excel.Range).Value;

            if (Node0Sonuç > Node1Sonuç)
            {
                return false;
            }
            else
            {
                return true;
            }
        }

        public static bool NaiveBayes(Microsoft.Office.Interop.Excel.Worksheet
worksheet_NaiveBayes, Microsoft.Office.Interop.Excel.Worksheet
worksheet_StandardizeDiyabetVeriSeti)
        {
            worksheet_NaiveBayes.Cells[18, 2] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 1];
            worksheet_NaiveBayes.Cells[18, 3] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 2];
            worksheet_NaiveBayes.Cells[18, 4] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 3];
            worksheet_NaiveBayes.Cells[18, 5] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 4];
            worksheet_NaiveBayes.Cells[18, 6] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 5];
            worksheet_NaiveBayes.Cells[18, 7] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 6];
            worksheet_NaiveBayes.Cells[18, 8] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 7];
            worksheet_NaiveBayes.Cells[18, 9] =
worksheet_StandardizeDiyabetVeriSeti.Cells[HastaBilgileri.mevcutHastaNumarası, 8];


            double pNaiveSağlıklıK = (double)(worksheet_NaiveBayes.Cells[20, 2] as
Microsoft.Office.Interop.Excel.Range).Value;
            double pNaiveHastaK = (double)(worksheet_NaiveBayes.Cells[24, 2] as
Microsoft.Office.Interop.Excel.Range).Value;
            double pNaiveSağlıklıE = (double)(worksheet_NaiveBayes.Cells[28, 2] as
Microsoft.Office.Interop.Excel.Range).Value;
            double pNaiveHastaE = (double)(worksheet_NaiveBayes.Cells[32, 2] as
Microsoft.Office.Interop.Excel.Range).Value;
```

```
            if (HastaBilgileri.cinsiyet == (int)HastaBilgileri.CinsiyetEnum.Kadın)
            {
                if (pNaiveSağlıklıK > pNaiveHastaK)
                {
                    return false;
                }
                else
                {
                    return true;
                }
            }

            else if (HastaBilgileri.cinsiyet == (int)HastaBilgileri.CinsiyetEnum.Erkek)
            {
                if (pNaiveSağlıklıE > pNaiveHastaE)
                {
                    return false;
                }
                else
                {
                    return true;
                }
            }
            return false;
        }
    }
    }
```

Form1

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Excel = Microsoft.Office.Interop.Excel;

namespace DiyabetArayüz
{

    public partial class Form_DiyabetTeşhisi : Form
    {
```

```
        public Form_DiyabetTeşhisi()
        {
            InitializeComponent();

            if (checkBoxTCNo.Checked == true)
            {
                HastaBilgileri.TCNo_Aktif = true;
                textBoxTCNo.Enabled = true;
            }
            else if (checkBoxTCNo.Checked == false)
            {
                HastaBilgileri.TCNo_Aktif = false;
                textBoxTCNo.Enabled = false;
            }
        }

        private void button_RastgeleAğaçHesapla_Click(object sender, EventArgs e)
        {
            if (hastaBilgileriniOku() == false)
                return;

            string myPath = Properties.Settings.Default.HastaKayıtları;
            Excel.Application excel = new Excel.Application();
            Excel.Workbook workbook = excel.Workbooks.Open(myPath);
            Excel.Worksheet worksheet_DiyabetVeriSeti =
workbook.Sheets["DiyabetVeriSeti"];
            Excel.Range userRange = worksheet_DiyabetVeriSeti.UsedRange;
            int satırSayısı = userRange.Rows.Count;

            Excel.Worksheet worksheet_YeniKayıt = workbook.Sheets["YeniKayıt"];
            Excel.Range userRange_YeniKayıt = worksheet_YeniKayıt.UsedRange;
            int satırSayısı_YeniKayıt = userRange_YeniKayıt.Rows.Count;

            if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_DiyabetVeriSeti,
satırSayısı) == true)
            {
                Excel.Worksheet worksheet_StandardizeDiyabetVeriSeti =
workbook.Sheets["StandardizeDiyabetVeriSeti"];

ExcelFonksiyonları.standardizeKayıtOku(worksheet_StandardizeDiyabetVeriSeti);
                ExcelFonksiyonları.excelKaydetmedenKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_StandardizeDiyabetVeriSeti,
worksheet_YeniKayıt);
                sonuçPenceresiAç("Rastgele ağaç",
SınıflandırmaAlgoritmaları.rastgeleAğaçHesapla());
            }
            else if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_YeniKayıt,
satırSayısı_YeniKayıt) == true)
            {
```

```
            Excel.Worksheet worksheet_StandardizeYeniKayıt =
workbook.Sheets["StandardizeYeniKayıt"];

ExcelFonksiyonları.standardizeKayıtOku(worksheet_StandardizeYeniKayıt);
            ExcelFonksiyonları.excelKaydetmedenKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_StandardizeYeniKayıt, worksheet_YeniKayıt);
            sonuçPenceresiAç("Rastgele ağaç",
SınıflandırmaAlgoritmaları.rastgeleAğaçHesapla());
          }
          else
          {
            MessageBox.Show("Hasta bilgisi bulunamadı", "Hata..",
MessageBoxButtons.OK, MessageBoxIcon.Error);
          }
      }

      private void button_ÇokKatmanlıAlgılayıcıHesapla_Click(object sender,
EventArgs e)
      {
        if (hastaBilgileriniOku() == false)
          return;

        string myPath = Properties.Settings.Default.HastaKayıtları;
        Excel.Application excel = new Excel.Application();
        Excel.Workbook workbook = excel.Workbooks.Open(myPath);
        Excel.Worksheet worksheet_DiyabetVeriSeti =
workbook.Sheets["DiyabetVeriSeti"];
        Excel.Range userRange = worksheet_DiyabetVeriSeti.UsedRange;
        int satırSayısı = userRange.Rows.Count;

        Excel.Worksheet worksheet_YeniKayıt = workbook.Sheets["YeniKayıt"];
        Excel.Range userRange_YeniKayıt = worksheet_YeniKayıt.UsedRange;
        int satırSayısı_YeniKayıt = userRange_YeniKayıt.Rows.Count;

        if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_DiyabetVeriSeti,
satırSayısı) == true)
        {
          Excel.Worksheet worksheet_StandardizeDiyabetVeriSeti =
workbook.Sheets["StandardizeDiyabetVeriSeti"];
          Excel.Worksheet worksheet_ÇokKatmanlıAlgılayıcı =
workbook.Sheets["ÇokKatmanlıAlgılayıcı"];

          bool sonuç_ÇokKatmanlıAlgılayıcı =
SınıflandırmaAlgoritmaları.ÇokKatmanlıAlgılayıcı(worksheet_ÇokKatmanlıAlgılayıcı,
worksheet_StandardizeDiyabetVeriSeti);
          sonuçPenceresiAç("Çok katmanlı algılayıcı", sonuç_ÇokKatmanlıAlgılayıcı);
```

```
            ExcelFonksiyonları.excelKaydetVeKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_ÇokKatmanlıAlgılayıcı,
worksheet_StandardizeDiyabetVeriSeti, worksheet_YeniKayıt);
            return;
        }
        else if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_YeniKayıt,
satırSayısı_YeniKayıt) == true)
            {
              Excel.Worksheet worksheet_StandardizeYeniKayıt =
workbook.Sheets["StandardizeYeniKayıt"];
              Excel.Worksheet worksheet_ÇokKatmanlıAlgılayıcı =
workbook.Sheets["ÇokKatmanlıAlgılayıcı"];

              bool sonuç_ÇokKatmanlıAlgılayıcı =
SınıflandırmaAlgoritmaları.ÇokKatmanlıAlgılayıcı(worksheet_ÇokKatmanlıAlgılayıcı,
worksheet_StandardizeYeniKayıt);
              sonuçPenceresiAç("Çok katmanlı algılayıcı",
sonuç_ÇokKatmanlıAlgılayıcı);

              ExcelFonksiyonları.excelKaydetVeKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_ÇokKatmanlıAlgılayıcı,
worksheet_StandardizeYeniKayıt, worksheet_YeniKayıt);
              return;
            }
          else
          {
            MessageBox.Show("Hasta bilgisi bulunamadı", "Hata..",
MessageBoxButtons.OK, MessageBoxIcon.Error);
          }
      }

      private void buttonNaiveBayesHesapla_Click(object sender, EventArgs e)
      {
        if (hastaBilgileriniOku() == false)
          return;

        string myPath = Properties.Settings.Default.HastaKayıtları;
        Excel.Application excel = new Excel.Application();
        Excel.Workbook workbook = excel.Workbooks.Open(myPath);
        Excel.Worksheet worksheet_DiyabetVeriSeti =
workbook.Sheets["DiyabetVeriSeti"];
        Excel.Range userRange = worksheet_DiyabetVeriSeti.UsedRange;
        int satırSayısı = userRange.Rows.Count;

        Excel.Worksheet worksheet_YeniKayıt = workbook.Sheets["YeniKayıt"];
        Excel.Range userRange_YeniKayıt = worksheet_YeniKayıt.UsedRange;
        int satırSayısı_YeniKayıt = userRange_YeniKayıt.Rows.Count;
```

```
            if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_DiyabetVeriSeti,
satırSayısı) == true)
            {
                Excel.Worksheet worksheet_StandardizeDiyabetVeriSeti =
workbook.Sheets["StandardizeDiyabetVeriSeti"];
                Excel.Worksheet worksheet_NaiveBayes = workbook.Sheets["NaiveBayes"];

                bool sonuç_NaiveBayes =
SınıflandırmaAlgoritmaları.NaiveBayes(worksheet_NaiveBayes,
worksheet_StandardizeDiyabetVeriSeti);
                sonuçPenceresiAç("Naive Bayes", sonuç_NaiveBayes);

                ExcelFonksiyonları.excelKaydetVeKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_NaiveBayes,
worksheet_StandardizeDiyabetVeriSeti, worksheet_YeniKayıt);
            }
            else if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_YeniKayıt,
satırSayısı_YeniKayıt) == true)
            {
                Excel.Worksheet worksheet_StandardizeYeniKayıt =
workbook.Sheets["StandardizeYeniKayıt"];
                Excel.Worksheet worksheet_NaiveBayes = workbook.Sheets["NaiveBayes"];

                bool sonuç_NaiveBayes =
SınıflandırmaAlgoritmaları.NaiveBayes(worksheet_NaiveBayes,
worksheet_StandardizeYeniKayıt);
                sonuçPenceresiAç("Naive Bayes", sonuç_NaiveBayes);

                ExcelFonksiyonları.excelKaydetVeKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_NaiveBayes, worksheet_StandardizeYeniKayıt,
worksheet_YeniKayıt);
            }
            else
            {
                MessageBox.Show("Hasta bilgisi bulunamadı", "Hata..",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }

        private void buttonKarmaYöntemHesapla_Click(object sender, EventArgs e)
        {
            if (hastaBilgileriniOku() == false)
                return;

            string myPath = Properties.Settings.Default.HastaKayıtları;
            Excel.Application excel = new Excel.Application();
            Excel.Workbook workbook = excel.Workbooks.Open(myPath);
            Excel.Worksheet worksheet_DiyabetVeriSeti =
workbook.Sheets["DiyabetVeriSeti"];
```

```
            Excel.Range userRange = worksheet_DiyabetVeriSeti.UsedRange;
            int satırSayısı = userRange.Rows.Count;

            Excel.Worksheet worksheet_YeniKayıt = workbook.Sheets["YeniKayıt"];
            Excel.Range userRange_YeniKayıt = worksheet_YeniKayıt.UsedRange;
            int satırSayısı_YeniKayıt = userRange_YeniKayıt.Rows.Count;

            if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_DiyabetVeriSeti,
satırSayısı) == true)
            {
                Excel.Worksheet worksheet_ÇokKatmanlıAlgılayıcı =
workbook.Sheets["ÇokKatmanlıAlgılayıcı"];
                Excel.Worksheet worksheet_StandardizeDiyabetVeriSeti =
workbook.Sheets["StandardizeDiyabetVeriSeti"];
                Excel.Worksheet worksheet_NaiveBayes = workbook.Sheets["NaiveBayes"];


ExcelFonksiyonları.standardizeKayıtOku(worksheet_StandardizeDiyabetVeriSeti);

                bool sonuç_RastgeleAğaç =
SınıflandırmaAlgoritmaları.rastgeleAğaçHesapla();
                bool sonuç_ÇokKatmanlıAlgılayıcı =
SınıflandırmaAlgoritmaları.ÇokKatmanlıAlgılayıcı(worksheet_ÇokKatmanlıAlgılayıcı,
worksheet_StandardizeDiyabetVeriSeti);
                bool sonuç_NaiveBayes =
SınıflandırmaAlgoritmaları.NaiveBayes(worksheet_NaiveBayes,
worksheet_StandardizeDiyabetVeriSeti);

                ExcelFonksiyonları.excelKaydetVeKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_NaiveBayes,
worksheet_StandardizeDiyabetVeriSeti, worksheet_ÇokKatmanlıAlgılayıcı);
                bool sonuç_KarmaYöntem = (sonuç_RastgeleAğaç &&
(sonuç_ÇokKatmanlıAlgılayıcı || sonuç_NaiveBayes)) || (sonuç_ÇokKatmanlıAlgılayıcı
&& sonuç_NaiveBayes);

                sonuçPenceresiAç("Karma Yöntem", sonuç_KarmaYöntem);
            }

            else if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_YeniKayıt,
satırSayısı_YeniKayıt) == true)
                {
                    Excel.Worksheet worksheet_ÇokKatmanlıAlgılayıcı =
workbook.Sheets["ÇokKatmanlıAlgılayıcı"];
                    Excel.Worksheet worksheet_StandardizeYeniKayıt =
workbook.Sheets["StandardizeYeniKayıt"];
                    Excel.Worksheet worksheet_NaiveBayes =
workbook.Sheets["NaiveBayes"];
```

```
ExcelFonksiyonları.standardizeKayıtOku(worksheet_StandardizeYeniKayıt);

            bool sonuç_RastgeleAğaç =
SınıflandırmaAlgoritmaları.rastgeleAğaçHesapla();
            bool sonuç_ÇokKatmanlıAlgılayıcı =
SınıflandırmaAlgoritmaları.ÇokKatmanlıAlgılayıcı(worksheet_ÇokKatmanlıAlgılayıcı,
worksheet_StandardizeYeniKayıt);
            bool sonuç_NaiveBayes =
SınıflandırmaAlgoritmaları.NaiveBayes(worksheet_NaiveBayes,
worksheet_StandardizeYeniKayıt);

            ExcelFonksiyonları.excelKaydetVeKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_NaiveBayes, worksheet_StandardizeYeniKayıt,
worksheet_ÇokKatmanlıAlgılayıcı, worksheet_YeniKayıt);
            bool sonuç_KarmaYöntem = (sonuç_RastgeleAğaç &&
(sonuç_ÇokKatmanlıAlgılayıcı || sonuç_NaiveBayes)) || (sonuç_ÇokKatmanlıAlgılayıcı
&& sonuç_NaiveBayes);

            sonuçPenceresiAç("Karma Yöntem", sonuç_KarmaYöntem);
        }



                else
        {
            MessageBox.Show("Hasta bilgisi bulunamadı", "Hata..",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    private void kaydetToolStripMenuItem_Click(object sender, EventArgs e)
    {
        if (hastaBilgileriniOku() == false)
            return;

        string myPath = Properties.Settings.Default.HastaKayıtları;
        string myPathTemp =
Path.GetDirectoryName(Path.GetDirectoryName(System.IO.Directory.GetCurrentDirec
tory())) + "HastaKayıtlarıTemp.xlsx";

        if (File.Exists(myPath) == false)
        {
            MessageBox.Show("Seçilen hasta kayıtlarını kontrol ediniz.", "Hata..",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }
```

```
            Excel.Application excel = new Excel.Application();
            Excel.Workbook workbook = excel.Workbooks.Open(myPath);
            Excel.Worksheet worksheet_DiyabetVeriSeti =
workbook.Sheets["DiyabetVeriSeti"];
            Excel.Range userRange = worksheet_DiyabetVeriSeti.UsedRange;
            int satırSayısı = userRange.Rows.Count;

            if (ExcelFonksiyonları.hastaBilgileriMevcutMu(worksheet_DiyabetVeriSeti,
satırSayısı) == true)
            {
                ExcelFonksiyonları.excelKaydetmedenKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti);
                MessageBox.Show("Hasta bilgisi zaten mevcuttur.", "Kaydetme işlemi..",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }

            if (HastaBilgileri.TCNo_Aktif == false)
            {
                ExcelFonksiyonları.excelKaydetmedenKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti);
                MessageBox.Show("TC Kimlik No bilgisi girilmelidir.", "Kaydetme işlemi..",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }

            Excel.Worksheet worksheet_YeniKayıt = workbook.Sheets["YeniKayıt"];
            Excel.Worksheet worksheet_EğitimVeriSeti =
workbook.Sheets["EğitimVeriSeti"];
            Excel.Worksheet worksheet_StandardizeYeniKayıt =
workbook.Sheets["StandardizeYeniKayıt"];
            Excel.Range userRange_YeniKayıt = worksheet_YeniKayıt.UsedRange;
            int satırSayısı_YeniKayıt = userRange_YeniKayıt.Rows.Count;
            int yeniSatırNumarası_YeniKayıt = satırSayısı_YeniKayıt + 1;
            ExcelFonksiyonları.yeniHastaEkle(worksheet_YeniKayıt,
yeniSatırNumarası_YeniKayıt);

            Excel.Worksheet worksheet_StandardizeDiyabetVeriSeti =
workbook.Sheets["StandardizeDiyabetVeriSeti"];
            ExcelFonksiyonları.standardizeKayıtEkle(worksheet_EğitimVeriSeti,
worksheet_StandardizeYeniKayıt, yeniSatırNumarası_YeniKayıt);
            ExcelFonksiyonları.excelKaydetVeKapat(workbook, excel, userRange,
worksheet_DiyabetVeriSeti, worksheet_YeniKayıt, worksheet_EğitimVeriSeti,
worksheet_StandardizeYeniKayıt, worksheet_StandardizeDiyabetVeriSeti);

            MessageBox.Show("Hasta bilgisi başarıyla eklenmiştir.", "Kaydetme işlemi..",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
```

```
private void ayarlarToolStripMenuItem_Click(object sender, EventArgs e)
{
    OpenFileDialog file = new OpenFileDialog();
    file.Filter = "Excel Dosyası |*.xls;*.xlsx;*.xlsm";
    file.FilterIndex = 2;
    file.RestoreDirectory = true;
    file.CheckFileExists = false;
    file.Title = "Hasta Kayıtlarını Seçiniz..";
    file.Multiselect = false;

    if (file.ShowDialog() == DialogResult.OK)
    {
        Properties.Settings.Default.HastaKayıtları = file.FileName;
        Properties.Settings.Default.Save();
    }
}

public bool hastaBilgileriniOku()
{
    try
    {
        if (comboBoxCinsiyet.Text == "Erkek")
            HastaBilgileri.cinsiyet = (int)HastaBilgileri.CinsiyetEnum.Erkek;
        else if (comboBoxCinsiyet.Text == "Kadın")
            HastaBilgileri.cinsiyet = (int)HastaBilgileri.CinsiyetEnum.Kadın;

        HastaBilgileri.trigliserit = Double.Parse(textBoxTrigliserit.Text);
        HastaBilgileri.glikoz = Double.Parse(textBoxGlikoz.Text);
        HastaBilgileri.ldlkolesterol = Double.Parse(textBoxLDLKolesterol.Text);
        HastaBilgileri.kolesterol = Double.Parse(textBoxKolesterol.Text);
        HastaBilgileri.hct = Double.Parse(textBoxHCT.Text);
        HastaBilgileri.mch = Double.Parse(textBoxMCH.Text);
        HastaBilgileri.eos = Double.Parse(textBoxEOS.Text);


        if (HastaBilgileri.TCNo_Aktif == true)
        {
            HastaBilgileri.TCNo = UInt64.Parse(textBoxTCNo.Text);
            if (Math.Floor(Math.Log10(HastaBilgileri.TCNo) + 1) == 11)
                return true;
            else
            {
                MessageBox.Show("Geçerli bir TC Kimlik No giriniz.", "Hata..",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                return false;
            }
        }

        return true;
```

```
            }
            catch
            {
                MessageBox.Show("Girilen hasta bilgilerini kontrol ediniz.", "Hata..",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                return false;
            }
        }

        public void sonuçPenceresiAç(string algoritma, bool sonuç)
        {
            if (sonuç == true)
            {
                MessageBox.Show(algoritma + " algoritmasına göre kişi hastadır", "Sonuç..",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            }

            if (sonuç == false)
            {
                MessageBox.Show(algoritma + " algoritmasına göre kişi hasta değildir",
"Sonuç..", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }

        private void checkBoxTCNo_CheckedChanged(object sender, EventArgs e)
        {
            if (checkBoxTCNo.Checked == true)
            {
                HastaBilgileri.TCNo_Aktif = true;
                textBoxTCNo.Enabled = true;
            }
            else if (checkBoxTCNo.Checked == false)
            {
                HastaBilgileri.TCNo_Aktif = false;
                textBoxTCNo.Enabled = false;
            }

        }
    }
}
```

EK-2 Eskişehir Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü 87914409-050.99 Sayılı ve 20.11.2020 Tarihli Etik Kurul Karar Belgesi

Eskişehir Teknik Üni. Evrak Tarih ve Sayısı: 08/12/2020-E.27803

T.C.
ESKİŞEHİR TEKNİK ÜNİVERSİTESİ REKTÖRLÜĞÜ
Hukuk Müşavirliği

Sayı : 87914409-050.99
Konu : 20.11.2020 tarihli 3/3 sayılı Etik Kurul Kararı

LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ MÜDÜRLÜĞÜNE

İlgi : 20/11/2020 tarihli ve 26447 sayılı yazı.

İlgi yazı ekinde Rektörlüğümüze gönderilen Prof. Dr. Nihal ERGİNEL'in danışmanlığını yaptığı ve Ezgi AKTAŞ'ın yazarlığını yaptığı **"Sınıflandırma Yöntemleri ve İstatiksel Analizler ile Hastalık Teşhisi"** başlıklı yüksek lisans tez çalışmasına ilişkin Fen ve Mühendislik Bilimler Bilimsel Araştırma ve Yayın Etiği Kurulu tarafından incelenmiş olup etik açıdan uygun bulunmuştur.

Bilgilerinizi ve uygulama dosyasının hazırlanmasında, ilgili kurumun, bulunması halinde Etik Kurulu Yönergesinin dikkate alınması konusunda gereğini rica ederim.

e-imzalıdır
Prof. Dr. Mustafa ŞENYEL
Fen Bilimleri ve Mühendislik Bilimleri Bilimsel Araştırma ve Yayın Etiği Kurulu Başkanı

ÖZGEÇMİŞ

Adı- Soyadı: Ezgi AKTAŞ POTUR

Yabancı Dil: İngilizce

Eğitim ve Mesleki Geçmişi:

- 2018, Eskişehir Osmangazi Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği
- 2019-2020, Eskişehir Teknik Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği Bölümü, Araştırma Görevlisi
- 2020-, Gazi Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği Bölümü, Araştırma Görevlisi

Yayınları:

- Aktaş Potur, E. ve Erginel, N. (2021). Predicting Survival of Heart Failure Patients via Classification Algorithms. 2nd International Conference on Access to Recent Advances in Engineering and Digitalization.

Ödülleri:

- 2018, Endüstri Mühendisliği Bölüm Birinciliği, Eskişehir Osmangazi Üniversitesi