T.C.

TRAKYA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

MAKİNE ÖĞRENMESİ ALGORİTMALARININ KARMAŞIKLIK VE DOYGUNLUK ANALİZİNİN BİR VERİ KÜMESİ ÜZERİNDE GERÇEKLEŞTİRİLMESİ

TOLGA DEMİRHAN

DOKTORA TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

Tez Danışmanı: Yrd. Doç. Dr. ÖZLEM UÇAR

EDİRNE-2015

T.Ü. Fen Bilimleri Enstitüsü onayı

Prof. Dr. Mustafa ÖZCAN
Fen Bilimleri Enstitüsü Müdürü

Bu tezin Doktora tezi olarak gerekli şartları sağladığını onaylarım.

Prof. Dr. Yılmaz KILIÇASLAN
Anabilim Dalı Başkanı

Bu tez tarafımca (tarafımızca) okunmuş, kapsamı ve niteliği açısından bir Doktora tezi olarak kabul edilmiştir.

[Ünvan, Ad ve Soyad yazın] Yrd. Doç.Dr. Özlem UÇAR
İkinci Tez Danışmanı (varsa) Tez Danışmanı

Bu tez, tarafımızca okunmuş, kapsam ve niteliği açısından Bilgisayar Mühendisliği Anabilim Dalında bir Doktora tezi olarak oy birliği ile kabul edilmiştir.

Jüri Üyeleri	İmza
Prof. Dr. Yılmaz KILIÇASLAN	
Prof. Dr. Nevcihan DURU	
Doç. Dr. Şaban AKTAŞ	
Yrd. Doç. Dr. Özlem UÇAR	
Yrd. Doç. Dr. Erdinç UZUN	

Tarih: 02/10/2015

T.Ü. FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ DOKTORA PROGRAMI

DOĞRULUK BEYANI

İlgili tezin akademik ve etik kurallara uygun olarak yazıldığını ve kullanılan tüm literatür bilgilerinin kaynak gösterilerek ilgili tezde yer aldığını beyan ederim.

02/10/2015

TOLGA DEMİRHAN



Doktora Tezi
Makine Öğrenmesi Algoritmalarının Karmaşıklık ve Doygunluk Analizinin Bir Veri Kümesi Üzerinde Gerçekleştirilmesi
T.Ü. Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

ÖZET

Bu çalışmada makine öğrenmesi algoritmaları kullanılarak eğitim alanında bir veri kümesi üzerinde doygunluk ve karmaşıklık analizi gerçekleştirilmiştir. Toplanan veriler arasında anlamsız ve eksik bulunan veriler temizlenerek 570 örneğe sahip bir veri kümesi elde edilmiştir. Sontest özniteliğinde yapılan veri dönüştürme işlemi ile 21 sınıflı, 5 sınıflı ve 2 sınıflı veri kümeleri elde edilmiştir. Weka'nın sahip olduğu LWL, J48, JRIP, Part, LMT, Baggıng, Random Forest, IBK, MultiLayer Perceptron, Voted Perceptron, SMO, Naïve Bayes sınıflandırma algoritmaları veri kümeleri üzerinde çalıştırılmıştır. 21 ve 5 sınıflı sontest özniteliğine sahip veri kümelerinden elde edilen başarının rastlantısal olduğu ve veri kümelerinin dengesiz olduğu sonucuna ulaşılmıştır. 2 sınıflı sontest özniteliğine sahip veri kümesinde algoritmalar çalıştırılmış sadece Naïve Bayes ve Voted Perceptron algoritmalarında verinin örnekleme yoğunluğunun doygunluk seviyesine ulaştığı sonucu çıkarılmıştır. Veri kümelerinin karmaşıklığını belirlemek üzere IBK, SMO, Voted Perceptron, J48 ve Naïve Bayes algoritmaları 2 sınıflı sontest özniteliğine sahip veri kümesine uygulanmıştır. Karmaşıklık analizinde verinin lineer olduğu durumlarda başarılı sonuçlar veren bir algoritma olan Voted perceptron algoritması en iyi sonuçları vermiştir. Yapılan karmaşıklık deneylerinde farklı üs değerleri için algoritmanın lineerliği değiştirilmiş, üs değeri arttıkça doğru sınıflandırma oranının düşmesi kullanılan veri kümesinin lineer olduğunu göstermiştir. Sınıflandırma gücü yüksek IBK algoritması ve destek karar makineleri (SVM) ile yapılan deneylerde eğitim verisi ile aşırı uyum (overfitting) durumu ortaya çıkmıştır.

Yıl **:** 2015
Sayfa Sayısı **:** 138
Anahtar Kelimeler **:** Makina Öğrenmesi, Sınıflandırma Algoritmaları, Karmaşıklık ve Doygunluk Analizi

Doctoral Thesis

Performing a Complexity and Saturation Analysis of Machine Learning Algorithms on a Data Cluster

Trakya University Institute of Natural Sciences

Department of Computer Engineering

ABSTRACT

In this study, a complexity and saturation analysis was performed on a data cluster with a training area, using machine learning algorithms. Among the data collected, null and missing data were cleaned and a data cluster of 570 instances was formed. By transforming the data in the 'Sontest' attribute, data clusters with 21, 5 and 2 classes were acquired. LWL, J48, JRIP, Part, LMT, Bagging, Random Forest, IBK, MultiLayer Perceptron, Voted Perceptron, SMO, Naïve Bayes classification algorithms of Weka were run on the data clusters. It was concluded that the success achieved with the data clusters which have a 'sontest' attribute with 21 and 5 classes was arbitrary and the clusters were imbalanced. The algorithms were run on the data cluster with 2 classes and the analyses revealed that data sampling density reached the point of saturation only in Naïve Bayes and Voted Perceptron algorithms. In order to determine the complexity level, IBK, SMO, Voted Perceptron, J48 and Naïve Bayes algorithms were run on the data cluster which has a 'sontest' attribute with 2 classes. In the complexity analysis, the best results were acquired with the Voted perceptron algorithm, which outputs successful results with linear data. The linearity of the algorithm was altered for different exponents in the complexity analyses and the decrease in the correct classification rate with the increase in the exponent value showed that the data cluster was linear. Overfitting to training data was observed in the tests that were run with Support Vector Machines and IBK, which has a high classification power.

Year **:** 2015
Number of Pages **:** 138
Keywords **:** Machine Learning, Classification Algorithms, Complexity and Saturation Analysis

ÖNSÖZ

Çalışmalarım süresince rehberliğinden, eleştirilerinden ve cesaretlendirmelerinden dolayı, başta danışmanım Yrd. Doç. Dr. Özlem UÇAR olmak üzere Prof. Dr. Yılmaz KILIÇASLAN'a, Doç. Dr. Şaban AKTAŞ'a ve Doç. Dr. Erdem UÇAR'a teşekkür etmek istiyorum.

Ayrıca, eşsiz sabırlarından ve desteklerinden dolayı, değerli eşim Pınar Aydan DEMİRHAN ve oğlum Yiğit Uras DEMİRHAN'a teşekkürlerimi sunarım.

İÇİNDEKİLER

ÖZET i

ABSTRACT ii

SİMGELER VE KISALTMALAR viii

ŞEKİLLER LİSTESİ ix

TABLOLAR LİSTESİ xi

BÖLÜM 1 1

GİRİŞ 1

BÖLÜM 2 4

KAVRAMSAL ÇERÇEVE 4

2.1. Veri Madenciliği ve Makina Öğrenmesi 4

2.2. Makina Öğrenmesinde kullanılan Öğrenme Yöntemleri 6

2.2.1. Denetimli Öğrenme (Supervised Learning) 6

2.2.2. Denetimsiz Öğrenme (Unsupervised Learning) 6

2.2.3. Takviyeli Öğrenme (Reinforcement Learning) 7

2.3. Model Oluşturma 7

2.4. Makine Öğrenmesinde Kullanılan Algoritmalar 9

2.4.1. Bayes.Naïve Bayes 9

2.4.2. Functions.MultilayerPerceptron 12

2.4.3. Lazy.IBK 14

2.4.4. Lazy.LWL 17

2.4.5. Meta.Bagging 17

2.4.6. Rules.Jrip 18

2.4.7. Rules.Part 19

2.4.8. Trees.J48 20

2.4.9. Trees.LMT 22

2.4.10. Trees.RandomForest 23
2.4.11. Functions.VotedPerceptron (Seçimli Algılayıcı) 26
2.4.12. Functions.SMO (Ardışık Minimal Optimizasyon) 26
2.5. Sınıflandırma Algoritmalarının Karşılaştırılmasında Kullanılan Kriterler 27
2.5.1. Karışıklık Matrisi 28
2.5.2. Kappa İstatistiği 29
2.5.3. Model Başarım Ölçütleri 30
2.5.3.1. Doğru tahmin oranı 30
2.5.3.2. F-Ölçütü 30
2.5.4. Ortalama Karesel Hatanın Karekökü (RMSE) 31
2.6. Yetersiz Uyum (underfitting) ve Aşırı Uyum (overfitting) 32
2.7. Makine Öğrenmesi Konulu Literatür Araştırması 33
2.8. WEKA 37
BÖLÜM 3 44
MATERYAL ve YÖNTEM 44
3.1. Veri Toplama Araçlarının Hazırlanması, Verilerin Toplanması 44
3.2. Veri Kümesi Önişlem 53
3.2.1. Veri Gürültüsünün Temizlenmesi 53
3.2.2. Veri kümelerinin Hazırlanması 54
3.2.3. Veri Dönüştürme 54
3.3. Geliştirilen Program 56
3.3.1. Anket ve Test Sonuçlarını Aktar 57
3.3.2. Veritabanı İşlemleri 61
3.3.3. Veriseti Dosyalarını Oluştur 73
BÖLÜM 4 80
DENEYLER VE ANALİZ 80

4.1. Test ve anket sonuçlarının dijital ortama aktarılıp saklanması 80
4.2. 21 Sınıflı Sontest Özniteliğine Sahip Veri Kümesinin Analizi 82
4.3. 5 Sınıflı Sontest Özniteliğine Sahip Veri Kümesinin Analizi 85
4.4. 2 Sınıflı Sontest Özniteliğine Sahip Veri Kümesi Doygunluk Deneyleri 87
4.4.1. Naïve Bayes ile Doygunluk Deneyleri 87
4.4.2. LWL ile Doygunluk Deneyleri 89
4.4.3. J48 ile Doygunluk Deneyleri 91
4.4.4. Jrip ile Doygunluk Deneyleri 93
4.4.5. Part Algoritması ile Doygunluk Deneyleri 95
4.4.6. LMT Algoritması ile Doygunluk Deneyleri 96
4.4.7. Bagging Algoritması ile Doygunluk Deneyleri 98
4.4.8. RandomForest Algoritması ile Doygunluk Deneyleri 100
4.4.9. IBK Algoritması ile Doygunluk Deneyleri 102
4.4.10. MultilayerPerceptron Algoritması ile Doygunluk Deneyleri 104
4.4.11. VotedPerceptron Algoritması ile Doygunluk Deneyleri 105
4.4.12. SMO Algoritması ile Doygunluk Deneyleri 107
4.4.13. Öznitelik Dönüştürme İşleminin Doygunluğa Etkisinin İncelenmesi 109
4.4.14. Öznitelik Eksiltmenin Doygunluğa Etkisinin İncelenmesi 112
4.4.15. Doygunluk Deneyleri Analiz Sonuçları 117
4.5. Veri Kümesi Karmaşıklık Deneyleri 119
4.5.1. Voted Perceptron Karmaşıklık Deneyleri 120
4.5.2. IBK Karmaşıklık Deneyleri 121
4.5.3. SMO Karmaşıklık Deneyleri 122
4.5.4. J48 Karmaşıklık Deneyleri 123
4.5.5. Naïve Bayes Karmaşıklık Deneyleri 124
4.5.6. Karmaşıklık Deneyleri Sonuçları 125

BÖLÜM 5 126
SONUÇLAR 126
KAYNAKLAR 130
ÖZGEÇMİŞ 135
TEZ ÖĞRENCİSİNE AİT TEZ İLE İLGİLİ BİLİMSEL FAALİYETLER 136
EK-A İl Milli Eğitim Müdürlüğü İzin Belgesi 137

SİMGELER VE KISALTMALAR

ML	MACHINE LEARNING
MÖ	MAKİNE ÖĞRENMESİ
LMT	LOGISTIC MODEL TREE
LWL	LOCALLY WEIGHTED LEARNING
KS	KAPPA STATICS
RMSE	ROOT MEAN SQUARED EROR
SMO	SEQUENTİAL MİNİMAL OPTİMİZATİON

ŞEKİLLER LİSTESİ

Şekil 2.1 Çok Katmanlı Yapay Sinir Ağı Modeli 14
Şekil 2.2 Veri kümesi Örnekleri 16
Şekil 2.3 Sınıflandırma 17
Şekil 2.4 Bagging Yöntemi 18
Şekil 2.5 J48 Karar Ağacı 22
Şekil 2.6 Random Forest Model oluşturma Algoritması 25
Şekil 2.7 Yapay Sinir Ağları 26
Şekil 2.8 İki Lagrange çarpanı problemin tüm kısıtlamalarını karşılamak zorundadır. Eşitsizlik kısıtlaması Lagrange çarpanlarının kutu içinde kalmalarına neden olur. Lineer eşitlik kısıtlamaları çarpanların diagonal çizgi üzerinde kalmalarını sağlar. Buna karşın SMO'nun bir adımı bir diagonal çizgi parçasındaki en iyi hedef fonksiyonunu bulmak zorundadır. 27
Şekil 2.9 3 Farklı Model Kullanılarak Oluşturulmuş Sınıflandırma Örneği [54] 32
Şekil 2.10 Arff Dosyası Görünümü 38
Şekil 2.11 Weka Karşılama Ekranı 38
Şekil 2.12 Weka Önişlem Ekranı 39
Şekil 2.12 Weka Sınıflandırma Ekranı 40
Şekil 2.13 Weka Kümeleme Ekranı 41
Şekil 2.14 Weka Birliktelik Kuralları Ekranı 41
Şekil 2.15 Weka Özellik Seçimleri Ekranı 42
Şekil 2.16 Weka Görselleştirme Ekranı 43
Şekil 3.1 Arff Formatlı Veri kümesi Dosyası 54
Şekil 3.2 Program Arayüzü 56
Şekil 3.3 Anket ve Test Sonuç Bilgilerini Aktar 57
Şekil 3.4 Veritabanı İşlemleri 61
Şekil 3.5 Sontest özniteliği ile yaş özniteliği 72
Şekil 3.6 Veritabanından Sil 72
Şekil 3.7 Veriseti Dosyalarını Oluştur 73
Şekil 4.1 Data isimli Veritabanında Kullanılan Tabloları 81

Şekil 4.2 Veritabanına Ekle ve Hesapla 81
Şekil 4.3 ontest Tablosu Kayıtları 82
Şekil 4.4 21 Sınıflı Veri Kümesinin Naïve Bayes Algoritması Sonuçları 83
Şekil 4.5 J48 Algoritması Sonucu 84
Şekil 4.6 Naïve Bayes Algoritması Weka Programı Çıktısı 85
Şekil 4.7 J48 Algoritması Weka Programı Çıktısı 86
Şekil 4.8 Naïve Bayes Algoritması Mutluluk Grafiği 88
Şekil 4.9 LWL Algoritması Mutluluk Grafiği 90
Şekil 4.10 J48 Algoritması Mutluluk Grafiği 92
Şekil 4.11 Jrip Algoritması Mutluluk Grafiği 93
Şekil 4.12 Part Algoritması Mutluluk Grafiği 95
Şekil 4.13 LMT Algoritması Mutluluk Grafiği 97
Şekil 4.14 Bagging Algoritması Mutluluk Grafiği 99
Şekil 4.15 Bagging Algoritması Mutluluk Grafiği 101
Şekil 4.16 IBK Algoritması Mutluluk Grafiği 103
Şekil 4.17 MultilayerPerceptron Algoritması Mutluluk Grafiği 104
Şekil 4.18 VotedPerceptron Algoritması Mutluluk Grafiği 106
Şekil 4.19 SMO Algoritması Mutluluk Grafiği 108
Şekil 4.20 Naïve Bayes Algoritması Mutluluk Grafiği 111
Şekil 4.21 Örnekli Veri kümesinin InfoGainAttributeEval Algoritması Sonuç Listesi 113
Şekil 4.22 Naïve Bayes Algoritması Mutluluk Grafiği 114
Şekil 4.23 VotedPerceptron Algoritması Mutluluk Grafiği 116

TABLOLAR LİSTESİ

Tablo 2.1 14 Günlük Tablo 9
Tablo 2.2 14 Günlük Tablo 10
Tablo 2.3 14 Günlük Tablo 11
Tablo 2.4 Karışıklık Matrisi 28
Tablo 2.5 Kappa Değerleri 29
Table 2.6 Karışıklık Matrisi 31
Tablo 3.1 Not Dönüşüm 55
Tablo 3.2 Not Dönüşüm 55
Tablo 4.1 Naïve Bayes Sonucu 83
Tablo 4.2 21 Sınıflı Veri Kümesinin J48 Algoritması Sonuçları 84
Tablo 4.3 Naïve Bayes Algoritması Sonuçları 85
Tablo 4.4 J48 Algoritması Sonuçları 86
Tablo 4.5 Naïve Bayes Algoritması Doygunluk Analizi Sonuçları 87
Tablo 4.6 LWL Algoritması Doygunluk Analizi Sonuçları 89
Tablo 4.7 J48 Algoritması Doygunluk Analizi Sonuçları 91
Tablo 4.8 Jrip Algoritması Doygunluk Analizi Sonuçları 93
Tablo 4.9 Part Algoritması Doygunluk Analizi Sonuçları 95
Tablo 4.10 LMT Algoritması Doygunluk Analizi Sonuçları 96
Tablo 4.11 Bagging Algoritması Doygunluk Analizi Sonuçları 98
Tablo 4.12 RandomForest Algoritması Doygunluk Analizi Sonuçları 100
Tablo 4.13 IBK Algoritması Doygunluk Analizi Sonuçları 102
Tablo 4.14 MultilayerPerceptron Algoritması Doygunluk Analizi Sonuçları 104
Tablo 4.15 VotedPerceptrom Algoritması Doygunluk Analizi Sonuçları 105

Tablo 4.16 SMO Algoritması Doygunluk Analizi Sonuçları.....107
Tablo 4.17.Öznitelik Dönüşüm Tablosu.....109
Tablo 4.18 Naïve Bayes Algoritması Doygunluk Analizi Sonuçları.....110
Tablo 4.19 Naïve Bayes Algoritması Doygunluk Analizi Sonuçları.....114
Tablo 4.20 VotedPerceptron Algoritması Doygunluk Analizi Sonuçları.....115
Tablo 4.21 VotedPerceptron Algoritması Karmaşıklık Deneyleri Sonuçları.....120
Tablo 4.22 IBK Algoritması Karmaşıklık Deneyleri Sonuçları.....121
Tablo 4.23 SMO Algoritması Karmaşıklık Deneyleri Sonuçları.....122
Tablo 4.24 J48 Algoritması Karmaşıklık Deneyleri Sonuçları.....123
Tablo 4.25 Naïve Bayes Algoritması Karmaşıklık Deneyleri Sonuçları.....124

BÖLÜM 1

GİRİŞ

Bilişim ve iletişim teknolojilerindeki gelişme nedeniyle bilişim çağı olarak adlandırılan 1990'lardan günümüze geçen süre içinde internet kullanımı hızla yaygınlaşmıştır. Mobil teknolojilerin gelişmesi, iletişim ve bilişim altyapılarının yaygınlaşması sayesinde bilgiye erişimin sınırları ortadan kalkmaya başlamıştır. Gerek bireysel gerekse kurumsal internet ve mobil teknoloji kullanımının artması sürekli olarak büyüyen bir bilgi öbeği oluşturmaktadır. Toplumun her kesiminde günlük hayatı sürdürmek veya kurumsal işler yürütmek için veri ihtiyacı sürekli artmaktadır. İnternette yapılan her gezinti, arama motorlarında yapılan aramalar, sosyal medya paylaşımları gibi pek çok bilgi veri tabanlarında depolanmaktadır. Ayrıca mobil operatörlerin verileri, çeşitli sensörlerden ve konum belirleyici cihazlardan alınan bilgiler de bu veri tabanlarında depolanan verinin büyük bir süratle artmasına neden olmaktadır. Bütün bu verinin anlamlı ve işlenebilir bir hale dönüştürülmüş biçimine Büyük Veri (Big Data) adı verilmektedir.

Dünyanın önde gelen bilişim araştırma gruplarından Gartner'ın tanımına göre büyük veri yüksek hacim, yüksek hız ve yüksek çeşitlilik içeren bilgi varlıklarıdır ve gelişmiş karar verme, kendi içinde keşif ve süreç optimizasyonun imkân vermek için yeni işlem formları gerektirmektedir [1].

Büyük veriyi oluşturan parametreler beşe ayrılmaktadır. Çeşitlilik, hız, hacim, doğrulama ve değer büyük verinin bileşenleri olarak sayılmaktadır. Çok çeşitli cihazlardan elde edilen farklı dillerde ve tipte verinin birlikte değerlendirilebilmesi gerekmektedir. Sürekli artan bir hızla üretilen büyük verinin kullanılması için aynı hızda işlem sayısına ve çeşitliliğine de ihtiyaç duyulmaktadır. Veri büyüklüğü her geçen dakika artarken bu veri ile işlem yapabilecek, depolayabilecek ve birleştirebilecek teknolojilerin

de hızla geliştirilmesi gerekliliği doğmaktadır. Ayrıca bu verinin korunması, gizli veya görülebilir olmasının sağlanması da önemli bir konu olmaktadır. Büyük veri için tüm bu gerekliliklerden daha önemli olan şey verinin değerli olmasıdır. Her ne amaçla kullanılacak olursa olsun anında ulaşılamayacak bilgi değerli olmamaktadır. Hacmi sürekli artan, büyük veri tabanlarında toplanan bu bilgi doğru şekilde işlenip analiz edilmediğinde ve istenildiği anda ulaşılabilir olmadığında bilgi çöplüğünden farksız olacaktır [1].

Büyük veri kaynaklarından değerli bilgiyi bulup çıkarmak son zamanların en önemli problemlerinden biridir. Büyük ölçekteki veriler arasından bilgiye ulaşmak için kullanılan yöntem genel olarak veri madenciliği adını almaktadır. Türkçe literatürde veri madenciliği ifadesinin yanında veritabanlarında bilgi madenciliği, bilgi çıkarımı, veri arkeolojisi gibi ifadeler de kullanılmaktadır.

Veri madenciliği bankacılık, sigortacılık, borsa, perakendecilik, telekomünikasyon, pazarlama, sağlık sektörü gibi pek çok alanda kullanılmaktadır. Büyük verinin doğru bir şekilde sınıflandırılması ve analiz edilmesi ile şirketlerin stratejilerini belirlerken yararlanabilecekleri, risk yönetiminde ve gelişimlerinde kullanabilecekleri değerli bilgiler ortaya çıkacaktır.

Toplanan veri miktarı büyüdükçe ve toplanan verilerdeki karmaşıklık arttıkça, daha iyi çözümleme tekniklerine ihtiyaç duyulmaktadır. Sınıflandırma, veri madenciliği araştırmalarının önemli bir parçasıdır. Amaç, yeni bir nesnenin, belirli sınıflar içinde hangi sınıfa ait olup olmadığını belirleyecek bir sınıflayıcı oluşturmaktır [2].

Veri madenciliği değerli bilginin elde edilmesinde yararlanılan bir aşamadır. Veritabanı teknolojileri, makine öğrenmesi, örüntü tanıma, yapay sinir ağları, istatistik, uzaysal veri analizi, verilerin görselleştirilmesi gibi değişik disiplinlerden farklı tekniklerin kullanılmasıyla veritabanlarından bilgi madenciliği yapılabilmektedir. İstatistik, makine öğrenmesi ve veritabanı teknolojisi veri madenciliği için en büyük katkıyı sağlamaktadır [3].

Yapay zeka olarak adlandırılan ve insan gibi düşünen, karar veren bilgisayarların geliştirilmesi günümüzün popüler konularından biridir. Bir bilgisayarın veri tabanında bulunmayan bir problemle karşılaştığında daha önce karşılaştığı durumlardan çıkarımlarda bulunarak bir karar vermesi ve çözüm üretmesi için çeşitli teknikler geliştirilmektedir. Bu tekniklerden biri de makine öğrenmesidir. Makine öğrenmesi

sistemi belirli bir eğitim verisi ile eğiterek yeni durumlar için karar üretmesini sağlayan bir tekniktir [4].

Makine öğrenmesi yapay zeka alanının bir dalıdır. Makine öğreniminin ana odak noktası karmaşık örüntülerin tanınmasını otomatik bir şekilde öğrenmek ve veriye dayalı olacak şekilde doğru kararlar vermektir [5].

Makine öğrenmesinin uygulamaları arasında arama motorları, doğal dil işleme, bilgisayarlı görme, konuşma ve el yazısı tanıma, tıbbi tanı ve örüntü tanıma gibi uygulamalar sayılabilir.

Makine öğrenmesinden çok önemli iki kavram doygunluk ve karmaşıklık meselesidir. Doygunluk kavramı önemlidir çünkü eğitim verisinin yeterli örnek sayısına ulaşması ile ilgili fikir verir. Yeterli örnek sayısına ulaşıldığında daha iyi öğrenme gerçekleşmekte ve tahminler daha başarılı olmaktadır. Ancak ondan daha önemlisi karmaşıklıktır çünkü günümüzde bilişim dünyasıdaki en büyük problem büyük veri problemidir. Büyük veri problemi elimizdeki verinin çok fazla olmasından kaynaklı bu verinin pratikte analiz edilmesi sorunudur. Veriyi analiz ederken de bizim için en önemli konu matematiksel olarak verinin karmaşıklığıdır. Verinin karmaşıklığını sadece veri üzerinde matematiksel çalışma yaparak elde etmeye çalışılırsa zaman ve enerji olarak insani sınırlara ulaşılır. Ancak veri karmaşıklığını bir labaratuardaki mikroskop gibi kullanılabilir bir enstrumanla ölçebiliriz bu enstrüman yapay öğrenme algoritmalarıdır. Çalışmada bu algoritmaları kullanarak elimizdeki verinin karmaşıklığını düzeyini ölçmek istiyoruz. Bununda büyük veri analizine katkı sağlayacak bir çalışma olacağını düşünüyoruz.

Bu çalışmada makine öğrenmesi algoritmaları kullanılarak eğitim alanlı bir veri kümesi üzerinde doygunluk ve karmaşıklık analizinin gerçekleştirilmesi amaçlanmıştır.

Tezin organizasyonu şu sekildedir. Bundan sonraki bölümde makine öğrenmesi kavramı, deneylerde kullanılan makine öğrenmesi algoritma ve teknikleri, sınıflandırma algoritması karşılaştırma ölçütleri ve weka tanıtılır. Üçüncü bölümde, veri toplama süreci, veri kümesine uygulanan önişlem ve geliştirilen program yer almaktadır. Dördüncü bölüm, gerçekleştirilen deneyleri ve analizlerini sunar. Tez, beşinci bölümde sonuçların özeti ile bitirilmektedir.

BÖLÜM 2

KAVRAMSAL ÇERÇEVE

2.1. Veri Madenciliği ve Makina Öğrenmesi

Günümüzde birçok kaynaktan veri alıp, bu verileri veri tabanlarında saklayan kurumların amaçlarından biri de ham verileri bilgiye dönüştürmektir. Bu işlem yani veriyi bilgiye dönüştürme işlemi Veri Madenciliği olarak adlandırılmaktadır [6].

Etkin bir veri madenciliği uygulaması için farklı tipteki verileri ele alma, veri madenciliği algoritmasının etkinliği ve ölçeklenebilirliği, sonuçların yararlılık, kesinlik ve anlamlılık kıstaslarını sağlaması, keşfedilen kuralların çeşitli biçimlerde gösterimi, farklı ortamlarda yer alan veri üzerinde işlem yapabilme, gizlilik ve veri güvenliği özelliklerinin sağlanması gereklidir [6].

Erişilmek istenen verilerin doğru anda doğru sıra ile bulunması veri madenciliğinin kullanım amaçlarından biridir. Veri madenciliği bir veri tabanında saklanabilecek bütün veriler için kullanılabilir. Veri madenciliğinin kullanım alanları için aşağıdaki sektörler örnek olarak verilebilir [7].

- Bankacılık (Risk analizleri ve usulsüzlük tespiti).
- Pazarlama (Çapraz satış analizleri, müşteri segmentasyonu).
- Sigortacılık (Müşteri kaybı sebeplerinin belirlenmesi, usulsüzlüklerin önlenmesi).
- Telekomünikasyon(Hile tespiti, hatların yoğunluk tahminleri).
- Borsa (Hisse senedi fiyat tahmini, genel piyasa analizleri).
- Tıp (Tıbbi teşhis, uygun tedavi sürecinin belirlenmesi).
- Bilim ve Mühendislik (Ampirik veriler üzerinde modeller kurularak bilimsel ve teknik problemlerin çözümlenmesi).
- Endüstri (Kalite kontrol, lojistik).

Alpaydın, (2004) aktardığına göre veri madenciliğine konu olan verilerin elle işlenmesi, analizlerinin yapılması ve bunlardan bilginin çıkarılması çok zaman alan ve

maliyetli bir işlemdir. Bu ve benzer veri kümelerinden bilginin elde edilmesi için makina öğrenmesi algoritmalarından faydalanılmaktadır [8].

Makine öğrenmesi, veri madenciliği sürecinde bilgileri ortaya çıkartırken kullanılan veritabanı teknolojileri, istatistiksel yöntemler, algoritmalar, yapay zeka tekniklerinden birisidir [9].

Ayrıca makine öğrenmesi, bilgisayarların geçmiş bilgilerden elde edilen tecrübelerden yararlanarak, gelecekteki olayları tahmin etmesine ve modelleme yapmasına imkan veren bir yapay zeka alanıdır. Bilgisayarın bir olay ile ilgili bilgileri ve tecrübeleri öğrenerek gelecekte oluşacak benzeri olaylar hakkında kararlar verebilmesi ve problemlere çözümler üretebilmesi olarak da tanımlanabilir [9].

Veri madenciliği, çıkan bilgi ve bu bilgilerin değerlendirilmesi ile uğraşırken makine öğrenmesi bu bilgiyi çıkarmayı sağlayan teknikler, bu teknikleri kullanan bilgisayar programlarının kendilerini geliştirmesi, öğrenme metotlarını geliştirerek, tahminleri ya da tanımları en iyi şekilde, yüksek performans ile nasıl çıkarabileceği ile ilgilenmektedir [10].

Makine öğrenmesinde, tümevarım kullanılarak çıkarımlar yapılmaktadır. Yapılan çıkarımlar, geleceğe yönelik tahminde bulunmak ya da bir tanım yapmak için kullanılır. Veri, geleceğe yönelik bilgi tahmini için kullanılacaksa, eğitim ve test aşaması gerçekleştirilir. Eğitim aşamasında, belirli miktarda veri kullanılarak bir model oluşturulur. Kullanılan veri, eğitim verisi olarak isimlendirilir. Oluşturulan model, tüm veriyi temsil eder. Test aşamasında ise, eğitim aşaması sonucunda oluşan modele, test aşaması için ayrılan ya da ileride toplanacak olan veriler sunulur. Ortaya çıkan bilgi ise tahmin etmek için kullanılır [10].

Makine öğrenmesinin günlük hayatımızdaki kullanım alanları aşağıda listelenmiştir [7];

- Parmak izi tanıma sistemleri
- Göz taraması (İris) ile tanıma sistemleri
- Yüz tanıma sistemleri
- El yazısı veya imza tanıma sistemleri
- Tıbbi verileri tanımlamada kullanılan sistemler
- Metin ve Mail analizinde kullanılan sistemler

2.2. Makina Öğrenmesinde kullanılan Öğrenme Yöntemleri

Makine öğrenmesi algoritmaları, kullandıkları öğrenme yöntemine göre sınıflandırılır. Makine öğrenmesinde üç farklı öğrenme yöntemi vardır. Bunlar Denetimli Öğrenim (Supervised Learning), Denetimsiz Öğrenim (Unsupervised Learning), Takviyeli Öğrenim (Reinforcement Learning) dir [10].

2.2.1. Denetimli Öğrenme (Supervised Learning)

"Örnekten öğrenme olarak da anılan denetimli öğrenme (Supervised Learning) yönteminde, algoritma, veri kümesindeki örneklerin sahip olduğu giriş ve sonuç verilerinin eşleştiği örneklerden bir fonksiyon üretir. Oluşan sistemin amacı, verilen örneklerden hareket ederek her bir sınıfa ilişkin özelliklerin bulunması ve bu özelliklerin belirli kural cümleleri ile ifade edilmesidir. Bu amaçla, bütün veri kümesinden bir öğrenme kümesi ayrılır ve model bu küme üzerinden kurulur. Ayrılan test kümesi ile de doğruluğu araştırılır. Modelin doğruluğu yeterli görülüp kullanılmak istendiği takdirde yeni gelen örneklere model uygulanır ve o örneklerin hangi sınıfa ait olduğunu kullanılan model belirler [11].

Fonksiyonun çıktıları, eğri uydurma (regresyon) ya da girdi nesnesinin sınıf etiketlenmesi (sınıflama) ile tahmin edilebilir.

2.2.2. Denetimsiz Öğrenme (Unsupervised Learning)

Denetimsiz öğrenme yönteminde, veri kümesindeki örneklerde giriş verilerine karşılık gelen çıkış değerleri bulunmamaktadır. Bu yöntemde mevcut girişler kullanılarak bir patern bulunması amaçlanmaktadır.

Denetimsiz öğrenme teknikleri özellikle olasılık teorisine dayanan kümeleme işleminde kullanılır. Kümeleme farklı gruptaki benzer nesneleri sınıflandırmadır. Kümeleme; makine öğrenmesi, veri madenciliği, örüntü tanıma ve resim analizi gibi birçok alanda kullanılan istatistiksel veri analizi için genel bir tekniktir [12].

Denetimsiz öğrenmede sınıflar önceden belirli olmayıp, veri kümesindeki verilerin özelliklerine göre sınıfların oluşturulması söz konusu olmaktadır. Denetimsiz öğrenme sürecinin işlediği tekniklere kümeleme teknikleri örnek verilebilir [11.

2.2.3. Takviyeli Öğrenme (Reinforcement Learning)

Hagan, (2002) aktardığına göre Takviyeli öğrenme yönteminde, denetimli öğrenme yöntemine benzer şekilde; algoritmaya verilen bir skorla her ağ girdisi için uygun bir çıktı sağlanır. Bu tip öğrenme denetimli öğrenmeye göre daha az kullanılır. Daha çok sistem kontrol uygulamalarında kullanılır [13].

Genel olarak, makina öğrenmesi problemlerinde sistem, son durumuna dayanarak, eylemleri seçmek için bir hareket şekli öğrenmek zorundadır. Takviyeli öğrenme problemleri ise, sistemin doğru davranışın örneklerini asla görmediği bu görevlerin alt kümesidir. Bunun yerine sadece denemeleri için pozitif ve negatif ödüller elde eder [13]

Alpaydın, (2011) aktardığna göre bazı uygulamalarda, sistemin çıkışı bir hareketler dizisi şeklindedir. Böyle bir durumda tek bir hareket önemli değildir. Önemli olan hedefe ulaşmak için doğru hareketlerin diziliminin planlanmasıdır. Herhangi bir durumda en iyi hareket diye tanımlayabileceğimiz bir şey yoktur. Bir hareket eğer iyi bir planın parçası ise iyi olarak kabul edilir. Böyle bir durumda, Makine öğrenmesi programı yapılan planların iyiliğine değer biçebilecek ve geçmişteki iyi hareketlerden bir plan oluşturabilmeyi öğrenebilecektir. Böyle öğrenme türleri takviyeli öğrenme olarak adlandırılır [14].

2.3. Model Oluşturma

Veri Madenciliğinde kullanılan modeller, temel olarak tahmin edici (predictive) ve tanımlayıcı (descriptive) olmak üzere iki ana başlık altında incelenmektedir. Tahmin edici modeller ile tanımlayıcı modeller arasındaki fark kesin sınırlarla ayrılmamıştır. Tahmin edici modeller anlaşılabilir olduğu ölçüde tanımlayıcı model olarak, tanımlayıcı modeller de tahmin edici model olarak kullanılabilirler [15].

Tahmin edici modellerde, sonuçları bilinen verilerden hareket edilerek bir model geliştirilmesi ve kurulan bu modelden yararlanılarak sonuçları bilinmeyen veri kümeleri için sonuç değerlerin tahmin edilmesi amaçlanmaktadır. Örneğin bir banka önceki dönemlerde vermiş olduğu kredilere ilişkin gerekli tüm verilere sahip olabilir. Bu verilerde bağımsız değişkenler kredi alan müşterinin özellikleri, bağımlı değişken değeri ise kredinin geri ödenip ödenmediğidir. Bu verilere uygun olarak kurulan model, daha

sonraki kredi taleplerinde müşteri özelliklerine göre verilecek olan kredinin geri ödenip ödenmeyeceğinin tahmininde kullanılmaktadır [16].

Tanımlayıcı modellerde ise karar vermeye rehberlik etmede kullanılabilecek mevcut verilerdeki örüntülerin tanımlanması sağlanmaktadır. X/Y aralığında geliri ve iki veya daha fazla arabası olan çocuklu aileler ile çocuğu olmayan ve geliri X/Y aralığından düşük olan ailelerin satın alma örüntülerinin birbirlerine benzerlik gösterdiğinin belirlenmesi tanımlayıcı modellere bir örnektir [16].

Veri madenciliği modellerini gördükleri işlevlere göre [16];

- Sınıflama (Classificatiori) ve Regresyon (Regression),
- Kümeleme (Clustering),
- Birliktelik Kuralları (Association Rules) ve Ardışık Zamanlı Örüntüler (Sequentiai Patterns, olmak üzere üç ana başlık altında incelemek mümkündür. Sınıflama ve regresyon modelleri tahmin edici, kümeleme, birliktelik kuralları ve ardışık zamanlı Örüntü modelleri tanımlayıcı modellerdir [16].

Tanımlanan problem için en uygun modelin bulunabilmesi, olabildiğince çok sayıda modelin kurularak denenmesi ile mümkündür. Bu nedenle veri hazırlama ve model kurma aşamaları, en iyi olduğu düşünülen modele varılıncaya kadar yinelenen bir süreçtir. Model kuruluş süreci kullanılan öğrenme yöntemine göre de farklılık göstermektedir [17].

Örneğin sınıflandırma algoritmaları kullanılırken tüm veri kümesi öğrenme ve test kümesi olarak ayrılmalı; modelin verilerden öğrenerek oluşturulması öğrenme kümesi, doğruluğunun kontrolü ise test kümesi ile gerçekleştirilmelidir [11].

Tahmin etmede yararlanılan veri madenciliği yöntemleri içerisinde yaygın olarak kullanılan yöntemler, sınıflandırma ve regresyon modelleridir [18].

Sınıflandırma ve regresyon modellerinde kullanılan başlıca teknikler aşağıda listelenmiştir [16];

- Yapay Sinir Ağları
- Genetik Algoritmalar
- K-En Yakın Komsu
- Naïve-Bayes
- Çoklu Regresyon, Lojistik Regresyon
- Faktör ve Ayırma analizleri

- Karar Ağaçları Şeklinde sıralanabilir.

Tez çalışmasında sınıflandırma modellerine yer verilmiştir.

2.4. Makine Öğrenmesinde Kullanılan Algoritmalar

Veri kümelerinde bulunan verinin yapısına, sayısına ve içeriğine göre kullanılan algoritmaların farklı problemler için farklı başarı değerleri vermesi karşılaşılan durumlardandır. Bu nedenle problemler için kullanılabilecek en iyi algoritma budur diye genel bir şey söylemek mümkün değildir. Makine öğrenmesinde seçilen bir sınıflandırma algoritması çalıştırıldığında önce belirtilmiş bir eğitim verisi ile eğitilir. Daha sonra eğitilen bu algoritma test verisi ile test edilir. Buradaki amaç daha önceden eğitilmiş algoritmanın, test verisinde bulunan sonuçları tahmin edip edememesi durumunun incelenmesidir. Buda algoritmanın başarısının gözlemlenmesi anlamına gelmektedir [19].

Çalışmada weka 3.7. programının sunduğu sınıflandırma algoritmalarının tümü denenerek analiz edilmiştir. En yüksek performans gösteren oniki algoritmanın sonuçları karşılaştırılarak sıralanmıştır. Aşağıda bu algoritmalar kısaca açıklanmıştır.

2.4.1. Bayes.Naïve Bayes

Dunham, (2003) aktardığna göre Naïve Bayes, bir sınıflandırma probleminde tüm özniteliklerin sınıflandırmadaki katkısının eşit ve birbirinden bağımsız olduğu varsayımıyla, koşullu olasılık temeline dayanan basit bir sınıflandırma şeklidir. Buna göre her bağımsız özniteliğin sonuca katkısının analizi ile bir koşullu olasılık değeri belirlenir. Farklı özniteliklerin, sonuç üzerindeki etkilerinin birleşimi ile sınıflandırma işlemi gerçekleştirilir. Bu metoda ‘Naïve’ yani saf denmesinin sebebi veri kümesinde yer alan özniteliklerin birbirinden bağımsız olduğu varsayımıdır [20].

Naïve Bayes algoritması kullanılarak bir tahmini örnek üzerinde yapalım [19]. Elimizde aşağıdaki örneklere sahip bir veri seti olsun.

Tablo 2.1 14 Günlük Tablo

Days	Outlook	Temperature	Humidity	Wind	PlayTennis
Day 1	sunny	hot	high	FALSE	no
Day 2	sunny	hot	high	TRUE	no
Day 3	overcast	hot	high	FALSE	yes
Day 4	rainy	mild	high	FALSE	yes
Day 5	rainy	cool	normal	FALSE	yes
Day 6	rainy	cool	normal	TRUE	no
Day 7	overcast	cool	normal	TRUE	yes

Day 8	sunny	mild	high	FALSE	no
Day 9	sunny	cool	normal	FALSE	yes
Day 10	rainy	mild	normal	FALSE	yes
Day 11	sunny	mild	normal	TRUE	yes
Day 12	overcast	mild	high	TRUE	yes
Day 13	overcast	hot	normal	FALSE	yes
Day 14	rainy	mild	high	TRUE	no

Elimizde bulunan bu verilere göre 15. gün oluşabilecek bir olayın sonucunu tahmin etmeye çalışalım. Bu 15. Gün'de bulunan veri Outlook(Sunny), Temperature(Cool), Humidity(High), Wind(Strong) seklinde olsun ve tenis maçının oynanıp oynanmayacağını tahmin edelim. Naïve Bayes algoritmasının formülü aşağıdaki gibidir.

$$p(c|F_1, \dots, F_n) = \frac{p(C)\, p(F_1,\dots,F_n|C)}{p(F_1,\dots,F_n)} \quad (2\text{-}1)$$

Bir sonucun çıkma olasılığı o sonuca etkiyen tüm faktörlerin o sonucu sağlama olasılıklarının çarpımıdır. Şimdi yukarıdaki örnek için bu hesabı yapacak olursak.

Tablo 2.2 14 Günlük Tablo

Days	Outlook	Temperature	Humidity	Wind	PlayTennis
Day 1	sunny	hot	high	FALSE	no
Day 2	sunny	hot	high	TRUE	no
Day 3	overcast	hot	high	FALSE	yes
Day 4	rainy	mild	high	FALSE	yes
Day 5	rainy	cool	normal	FALSE	yes
Day 6	rainy	cool	normal	TRUE	no
Day 7	overcast	cool	normal	TRUE	yes
Day 8	sunny	mild	high	FALSE	no
Day 9	sunny	cool	normal	FALSE	yes
Day 10	rainy	mild	normal	FALSE	yes
Day 11	sunny	mild	normal	TRUE	yes
Day 12	overcast	mild	high	TRUE	yes
Day 13	overcast	hot	normal	FALSE	yes
Day 14	rainy	mild	high	TRUE	no

- PlayTennis sonucunun yes çıkması olasılığı verilen örneklere bakıldığında 9 tanesi yes olduğundan 9/14'dür.
- Outlook Sunny iken PlayTennis'in yes çıkma olasılığı: 2/9
- Temperature Cool iken PlayTennis'in yes çıkma olasılığı: 3/9
- Humidity High iken PlayTennis'in yes çıkma olasılığı: 3/9
- Wind True iken PlayTennis'in yes çıkma olasılığı: 3/9

Dolayısıyla sonucun Yes çıkma olasılığı P(y)=9/14 x 2/9 x 3/9 x 3/9 x 3/9=0,0053

Şimdi ise sonucun No çıkma olasılığını inceleyelim. PlayTennis sonucunun no çıkması olasılığı sonuçlardan 5 tanesi no olduğundan 5/14'dür. Bu olasılık için bundan sonraki tüm hesaplarımız bu 5 kayıt üzerinden yapılacaktır.

Tablo 2.3 14 Günlük Tablo

Days	Outlook	Temperature	Humidity	Wind	PlayTennis
Day 1	sunny	hot	high	FALSE	no
Day 2	sunny	hot	high	TRUE	no
Day 3	overcast	hot	high	FALSE	yes
Day 4	rainy	mild	high	FALSE	yes
Day 5	rainy	cool	normal	FALSE	yes
Day 6	rainy	cool	normal	TRUE	no
Day 7	overcast	cool	normal	TRUE	yes
Day 8	sunny	mild	high	FALSE	no
Day 9	sunny	cool	normal	FALSE	yes
Day 10	rainy	mild	normal	FALSE	yes
Day 11	sunny	mild	normal	TRUE	yes
Day 12	overcast	mild	high	TRUE	yes
Day 13	overcast	hot	normal	FALSE	yes
Day 14	rainy	mild	high	TRUE	no

Outlook Sunny iken PlayTennis'in no çıkma olasılığı: 3/5

Temperature Cool iken PlayTennis'in no çıkma olasılığı: 1/5

Humidity High iken PlayTennis'in no çıkma olasılığı: 4/5

Wind True iken PlayTennis'in no çıkma olasılığı: 3/5

Dolayısıyla sonucun No çıkma olasılığı

P(n)=5/14 x 3/5 x 1/5 x 4/5 x 3/5=0,0205

No çıkma olasılığı yes çıkma olasılığından büyük olduğundan dolayı (P(n)>P(y)) algoritmanın öğrendiği verilerden tahmini tenis maçının oynanmayacağı yönündedir [19].

Dunhom, (2003) aktardığına göre ‘Naïve Bayes’ algoritmasının birçok avantajı bulunur. Öncelikle kullanımı çok kolaydır. İkinci olarak ise diğer sınıflandırma algoritmalarının büyük çoğunluğunun aksine eğitim verisinin sadece bir kez taranması yeterlidir. Ayrıca kayıp (boş) değerler de olasılık hesaplarında yok sayılarak kolaylıkla ele alınabilmektedir. Basit ilişkilerin bulunduğu durumlarda çoğunlukla çok iyi sonuç veren bir tekniktir [20].

Dunhom, (2003) aktardığına göre basit kullanımına karşın ‘Naïve Bayes’ bazı durumlarda tatmin edici sonuçlar vermeyebilir. Öncelikle öznitelikler bağımsız

olmayabilir. Bu durumda öznitelik alt kümeleri kullanılabilir. İkinci olarak ise bu teknik, sürekli değerleri ele alamaz. Bu durumda da sürekli değerleri aralıklara bölmek gerekir. Bu tip sorunlarının yanı sıra ilişkili çözümleri olsa da bu çözümlerin uygulanması kolay olmamakla beraber bu çözümleri uygulama şekli de sonuçları ciddi ölçüde etkiler [20].

2.4.2. Functions.MultilayerPerceptron

MultilayerPerceptron (MLP), insan beyninin çalışma prensibini taklit eden yapay sinir ağları, tahmin amaçlı olarak da kullanılan önemli bir araçtır [21]. Günlük hayatta karşılaşılan problemlerin hemen hemen hepsi doğrusal olmayan nitelik taşımaktadır. Bu durum, yapay sinir ağları (YSA) ile ilgili yapılan bilimsel araştırmaların pek çoğunun durmasına neden olmuştur. Birkaç araştırmacının çalışmalarına devam ederek YSA nın doğrusal olmayan problemlerin de çözümünde kullanıldığını göstermeleri ile YSA ile ilgili çalışmalar tekrar hız kazanmıştır. Doğrusal olmayan bir ilişki gösteren XOR problemini çözmek amacıyla yapılan çalışmalar sonucunda Çok Katmanlı Algılayıcı (ÇKA) (Multilayer Perceptron) modeli geliştirilmiştir. Rumelhart ve arkadaşları tarafından geliştirilen bu modele hata yayma modeli veya geriye yayılım modeli (backpropogation network) de denilmektedir [22].

Çok Katmanlı Algılayıcı modeli yapay sinir ağlarına olan ilgiyi çok hızlı bir şekilde arttırmış ve YSA tarihinde yeni bir dönem başlatmıştır. Bu ağ modeli özellikle mühendislik uygulamalarında en çok kullanılan sinir ağı modeli olmuştur. Birçok öğretme algoritmasının bu ağı eğitmede kullanılabilir olması, bu modelin yaygın kullanılmasının sebebidir [22].

Tahmin amaçlı en yaygın kullanılan yapay sinir ağı tipi, çok katmanlı algılayıcıdır. Tipik bir yapay sinir ağı, nöronlardan oluşan katmanların bileşimidir. Çok katmanlı algılayıcı (ÇKA), dışarıdan verileri alan girdi katmanından, ağın çıktılarını dışarıya veren çıktı katmanından ve bu ikisi arasında genellikle bir ve bazen de daha fazla ara, gizli katmandan meydana gelmektedir [21].

Çok Katmanlı Algılayıcı ağı öğrenme stratejisi denetimli öğrenme yöntemidir. Kurulan Çok Katmanlı Algılayıcı ağına eğitim seti içerisinde örnek bilgiler verilirken aynı zamanda bu bilgilerin (girdilerin) karşılığında çıkması beklenen çıktı bilgileri de verilir. Çok Katmanlı Algılayıcı ağı öğrenme sürecinde kendisine verilen örneklerden genelleme yapar ve çözmesi gereken problem için bir çözüm yolu veya kalıbı oluşturur.

Bu çözüm yolu veya kalıba bakarak daha sonra Çok Katmanlı Algılayıcı ağına sunulan problemler için çözümler üretir. Çok Katmanlı Algılayıcı ağı kendisine verilen her girdi için bir çıktı vermek üzere programlanmıştır. Çok Katmanlı Algılayıcı en küçük kareler yöntemine dayalı Genelleştirilmiş Delta Kuralını kullanır [23].

Çok Katmanlı Algılayıcı Yapısı:

Bir Çok Katmanlı Algılayıcı modeli, aşağıdaki şekilde de gösterildiği gibi bir giriş, bir veya daha fazla ara ve bir de çıkış katmanından oluşur. Çok katmanlı ağlar ara katmanının olması nedeni ile tek katmanlı ağlardan ayrılır. Ara katman sayısı arttıkça girdi verilerinden edinilen istatistiki bilgi artmakta dolayısı ile öğrenme daha iyi olmaktadır. Ağın yapısına göre Çok Katmanlı Algılayıcı ileri beslemeli veya geri yayılımlı bir ağ olarak tasarlanabilir [23]. Çok Katmanlı Algılayıcı katmanları kısaca açıklayacak olursak;

Giriş Katmanı:

Giriş katmanı, verilerin ağa girişini sağlar. Bu katmanda herhangi bir işlem olmadan gelen bilgiler bir sonraki katmana aktarılır. Girdi sayısında bir kısıtlama yoktur. Ancak girdi katmanındaki işlem elemanlarının sadece bir girdisi olmaktadır. Girdi katmanının çıktısı ara katmandaki tüm işlem elemanlarına gönderilir. Kısaca girdi katmanındaki işlem elemanı girdinin diğer katmanlara dağıtılmasını sağlamaktadır [23].

Ara katman:

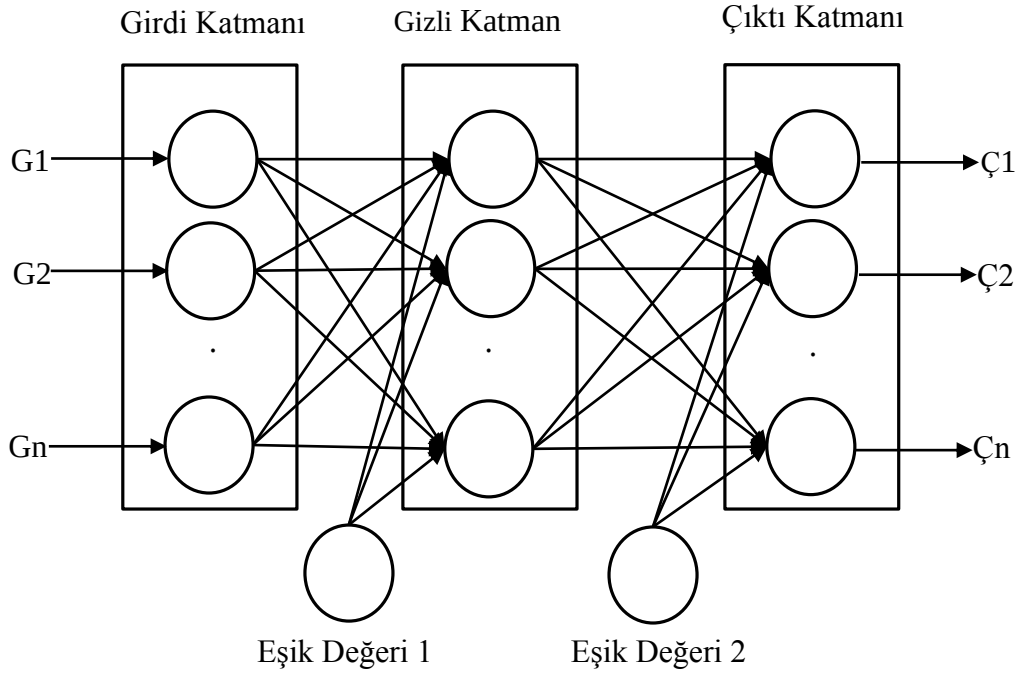
Saklı katman olarak da tanımlanmaktadır. Girdi katmanından gelen bilginin işlenmesi bu katmanda olur. Ara katman sayısı ve ara katmanlardaki işlem elemanı sayısı ise, deneme-yanılma yolu ile bulunur [22].

Öztemel, (2006) aktardığına göre işlemin yapılmasından sonra bilgi bir sonraki katmana aktarılır. Çok Katmanlı Algılayıcı ağında birden fazla ara katman ve her katmanda birden fazla işlem elemanı olabilir. Ara katmanda faaliyet gösteren işlem elemanları bir sonraki katmandaki tüm işlem elemanları ile bağlantılıdır. Yani oluşturduğu yeni bilgiyi bir sonraki katmandaki tüm işlem elemanlarına göndermektedir [23].

Çıkış Katmanı:

Öztemel, (2006) aktardığına göre Çıktı katmanı kendinden önceki ara katmanlardan gelen bilgileri işleyerek girdi katmanında girdi olarak verilen bilgilere karşılık çıktılar üretir. Çıktı katmanında da ara katmanda olduğu gibi birden fazla işlem elemanı olabilir. Çıkış katmanındaki eleman sayısı ise yine uygulanan probleme dayanılarak belirlenir. Katmanda bulunan tüm işlem elemanları ile bağlantılıdır. Ancak çıktı katmanındaki her işlem elemanının sadece bir tane çıktısı vardır [23].

Çok Katmanlı Algılayıcı modelinin temel amacı, ağın beklenen çıktısı ile ürettiği çıktı arasındaki hatayı en aza indirmektir. Bu ağlara eğitim sırasında hem girdiler hem de o girdilere karşılık üretilmesi gereken çıktılar gösterilir (danışmanlı öğrenme). Ağın görevi her girdi için o girdiye karşılık gelen çıktıyı üretmektir. Örnekler giriş katmanına uygulanır, ara katmanlarda işlenir ve çıkış katmanından da çıkışlar elde edilir. Kullanılan eğitme algoritmasına göre, ağın çıkışı ile arzu edilen çıkış arasındaki hata tekrar geriye doğru yayılarak hata minimuma düşünceye kadar ağın ağırlıkları değiştirilir [22].



Şekil 2.1 Çok Katmanlı Yapay Sinir Ağı Modeli [23]

2.4.3. Lazy.IBK

Özkan, (2008) aktardığna göre IBk, kNN algoritması olarak ta bilinir. En yakın komşu yöntemine “tek bağlantı kümeleme yöntemi” ismi de verilmektedir. İlk anda tüm

gözlem değerleri birer küme olarak değerlendirilir. Aşamalı olarak bu kümeler birleştirilerek yeni kümeler elde edilir. Bu yöntemde öncelikle gözlemler arasındaki mesafe belirlenir [24]. Güner, (2008) aktardığna göre K-en yakın komşu (k-nearest neighbour - kNN) algoritması benzerlik yoluyla öğrenme teknikleri kullanan algoritmalardan biridir. Weka'da kNN algoritmasına IBk adı verilmiştir [24].

,K En Yakın Komşu yöntemi, sınıflandırma problemini çözen denetimli öğrenme yöntemleri arasında yer alır. Yöntemde; sınıflandırma yapılacak verilerin öğrenme kümesindeki normal davranış verilerine benzerlikleri hesaplanarak; en yakın olduğu düşünülen k verinin ortalamasıyla, belirlenen eşik değere göre sınıflara atamaları yapılır. Önemli olan, her bir sınıfın özelliklerinin önceden net bir şekilde belirlenmiş olmasıdır. Yöntemin performansını k en yakın komşu sayısı, eşik değer, benzerlik ölçümü ve öğrenme kümesindeki normal davranışların yeterli sayıda olması kriterleri etkilemektedir [25].

Avantajları [26]

- Uygulanabilirliği basit bir algoritmadır.
- Gürültülü eğitim dokümanlarına karşı dirençlidir.
- Eğitim dokümanları sayısı fazla ise etkilidir.

Dezavantajları [26]

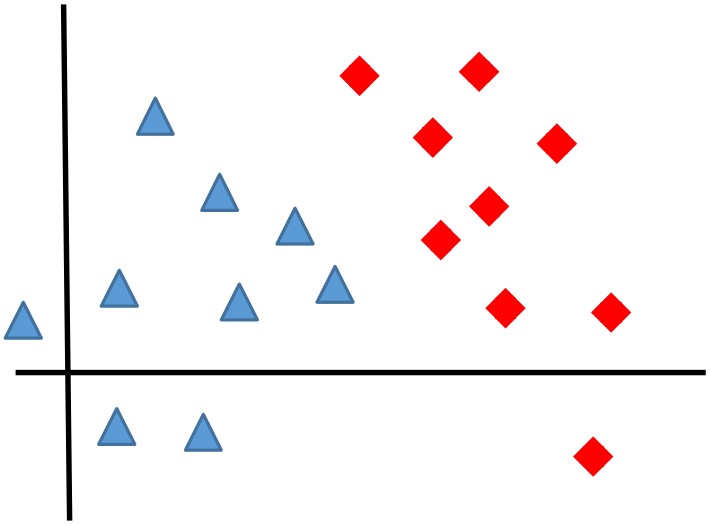
- K parametreye ihtiyaç duyar.
- Uzaklık bazlı öğrenme algoritması, en iyi sonuçları elde etmek için, hangi uzaklık tipinin ve hangi niteliğin kullanılacağı konusunda açık değildir.
- Hesaplama maliyeti gerçekten çok yüksektir çünkü her bir sorgu örneğinin tüm eğitim örneklerine olan uzaklığını hesaplamak gerekmektedir. Bazı indeksleme metodları ile (örneğin K-D ağacı), bu maliyet azaltılabilir.
- En yakın komşuluk prensibine dayanır. Tüm dokümanlar vektörel olarak temsil edilir. Sorgu dokümanı ile diğer dokümanlar arasındaki kosinüs benzerliği hesaplanır. Similarity oranı 1'e en yakın olan n tane vektörün kategorisinden çok olanı dokümana atanır.

K-NN algoritması adımları [27] ;

1. K değerinin belirlenmesi.

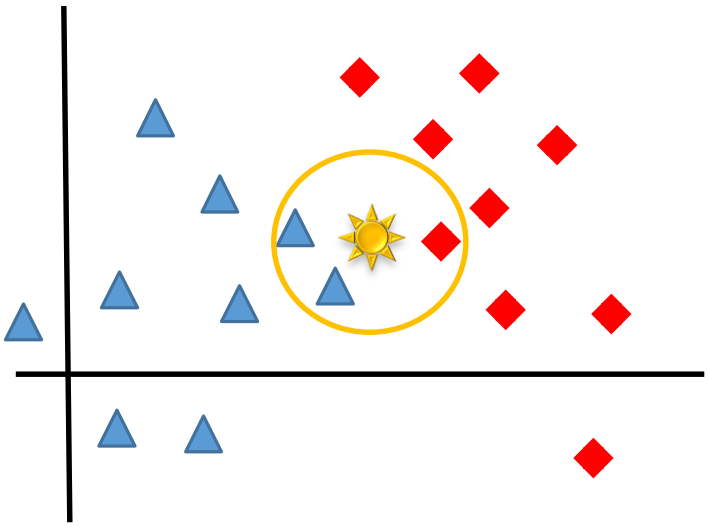
2. Tüm öğrenme örnekleri ile olan uzaklığının hesaplanması.
3. Minimum uzaklığa göre sıralama işleminin yapılması.
4. Uzaklık Fonksiyonları(Manhattan, Minkowski, Öklid uzaklık fonksiyonu Ait oldukları sınıf değerlerinin bulunması.
5. Baskın olan sınıf seçilir.

Bu adımları bir örnek üzerinde gösterelim. Burada k değerinin belirlenmesi, en yakın kaç vektöre bakılması gerektiğini ifade etmektedir. Örneğin k değerimiz 3 olsun. Bu durumda öğrenme kümesindeki dokümanlardan en yakın 3 tanesi alınarak dokümanın hangi sınıfa ait olduğuna karar verilir. Örneğin aşağıdaki gibi iki boyutlu koordinat sistemine yerleştirilmiş örnekler olsun [27].



Şekil 2.2 Veri kümesi Örnekleri

Yukarıdaki örneklerden maviye mi yoksa kırmızıya mı benzediğini tespit etmemiz gereken bir de sarı örneğimiz olsun. K değerimizi 3 aldığımızdan yeşil örneğe en yakın olan 3 örneğe bakarız. K değeri 4 olsaydı en yakın 4 örneğe bakacaktık. En yakın 3örnekten 2 tanesi mavi olduğundan sarı örnek için de mavi sınıfına aittir diyebiliriz [27].



Şekil 2.3 Sınıflandırma

2.4.4. Lazy.LWL

Frank et al., (2003) paylaştığına göre LWL (Locally Weighted Learning) diğer lazy öğrenme metodlarına benzerdir. Ancak yeni bir örnek sınıflandırılırken farklı davranır. LWL algoritması öğrenme örneklerinin küme ağırlığını kullanarak yeni bir Naïve Bayes modeli kurar. Bu deneysel olarak test edilen çoğu veri kümeleri üzerinde standart Naïve Bayes yanı sıra en yakın komşu yöntemlerini geride bırakır [28].

2.4.5. Meta.Bagging

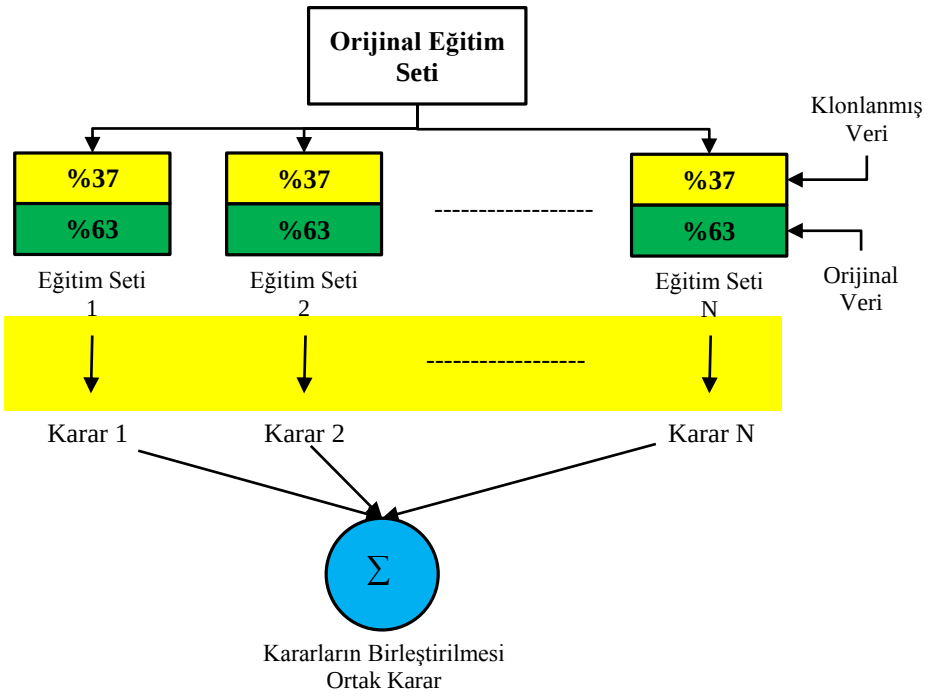
Bauer, E., Kohavi, R. (1998) ve Sutton, C. (2005), bagging algoritması ile ilgili şunları aktarıyorlar. Bagging sınıflama yöntemi 1996 yılında Leo Breiman tarafından bulmuştur. Genelde ağaç tabanlı sınıflandırıcılarda kullanılan bir yöntemdir. Bagging sınıflama algoritması, hem sınıflama istikrarını hem de doğruluğunu arttırmak amacıyla kullanılır. Çünkü bu sınıflama yöntemi tahmin gücünü arttıran genel bir tekniktir. Varyansı düşürücü etkisi vardır, böylece sınıflama hatasını azaltmaktadır. Aşırı öğrenmeye ve eksik gözlemlere karşı güçlü bir yapısı vardır. Bagging yöntemi, N çaplı eğitim setinden bootstrap örnekleme tekniği ile m adet n çaplı yeni eğitim setleri oluşturur ($n \leq N$). Bir bootstrap örneklemi, eğitim setinden yerine koyarak örnekleme yöntemiyle x adet örneğin seçilmesinden oluşur. m adet bootstrap örneklemi B1,…Bm üretilir ve her bir bootstrap örneklemi için Ci gibi m (ağaç sayısı) adet sınıflayıcı oluşturulur. Final sınıfı ise m adet sınıflayıcı içerisinde en çok oyu alan sınıftır. Her bir örnekleme yerine koyarak örnekleme yöntemiyle yapıldığı için bazı örnekler, birden fazla kez aynı örneklemde

bulunabilmektedir. Eğer n=N ise bu örneklemde orijinal eğitim setinin %63,2'sinin olması (farklı örnekler), geri kalanın ise bazı örneklerin tekrarından oluşması beklenir [29].

Breiman, L. (1996), da ağaç tabanlı sınıflandırıcılar için Bagging yönteminde veri, eğitim ve test veri seti olarak ayrıldığından bahseder. Eğitim setinden bootstrap örnekleme yöntemiyle m adet ağaç oluşturulur. Bu ağaçlarda dallara ayırıcı özellikteki değişken, tüm değişkenler arasından rastgele seçilir ve oylamaya tabi tutularak en yüksek oyu alan sınıf nihai (en son) sınıf olarak belirlenir [29].

Bagging'te temel öğreniciler eğitim verisine duyarlı ise başarılı sonuç vermesi beklenir [30].

Aşağıda bagging algoritmasının çalışması gösterilmiştir.



Şekil 2.4 Bagging Yöntemi [31]

2.4.6. Rules.Jrip

Witten and Frank, (2000) JRip, RIPPER algoritmasının WEKA implementasyonu olduğunu ifade eder. RIPPER'ın açılımı Repeated Incremental Pruning to Produce Error Reduction dır. Jrip (RIPPER), basit ve bilginin sunumu ve ifadesi kolay olduğu için karar ağaçları gibi popüler algoritmalardandır [32] .

Cohen tarafından (1995) yılında tasarlanmış olan algoritmadır. RIPPER özellikle gürültülü büyük veri kümelerinde daha etkilidir. Ripper algoritmasında iç ve dış olmak

üzere iki çeşit döngü bulunmaktadır. Dış döngü kural tabanına aynı anda bir kural ekler ve iç döngü geçerli kurala aynı anda bir koşul ekler. Bilgi kazanç değeri koşulların kurala

1. Ripper(Pos, Neg, k)
2. RuleSet ←LearnRuleSet(Pos, Neg)
3. k kere için
4. RuleSet←OptimizeRuleSet(RuleSet, Pos, Neg)
5. LearnRuleSet(Pos, Neg)
6. RuleSet ← Ø
7. DL←DescLen(RuleSet, Pos, Neg)
8. Tekrarla
9. Rule ← Öğren Rule(Pos, Neg)
10. Rule' u RuleSet'e ekle
11. DL'←DescLen(RuleSet, Pos, Neg)
12. Eğer DL'>DL+64 ise
13. PruneRuleSet(RuleSet,Pos, Neg)
14. RuleSet'e geri dön
15. Eğer DL1<DL ise DL←DL'
16. Pos ve Neg olarak dönüştürülmüş örnekleri sil
17. Pos=Ø olana kadar
18. RuleSet'e geri dön

eklenmesi ile maksimum olur. Bu işlem olumsuz örnek kalmayana kadar devam eder. Bu algoritmanın zaman karmaşıklığı ($N\log_2 N$) dir. Kural tabanı tanımı uzunluğu DL olarak adlandırılır. Algoritma aşağıdaki şekilde gösterilmiştir [33].

2.4.7. Rules.Part

PART (Projective Adaptive Resonance Theory), EIBE ve Witten tarafından önerilen böl ve yönet kural öğrenenidir. Algoritma karar listeleri olarak anılan düzenli kurallar kümesi üretir. Yeni veri listedeki her bir kural ile sırayla karşılaştırılır ve öğe eşleşen ilk kuralın kategorisine atanır (varsayılan olarak herhangi bir eşleşme yoksa uygulanır). Part her tekrarında kısmi C4.5 karar ağacı oluşturur ve "en iyi" yaprağı kural yapar. Part algoritması C4.5 ve RIPPER kural öğrenme algoritmalarının birleşimidir [34].

PART açılımı Projective Adaptive Resonance Theory dir. PART algoritması için giriş, tedbir ve mesafe parametreleridir.

Başlangıç: F1 katmanındaki düğüm sayısı m:= giriş katmanındaki boyut sayısı.

F katmanındaki düğüm sayısı m= her kümelenme düzeyinde oluşturulabilir kümelerin beklenen maksimum sayısı. Başlangıç parametreleri L, ρo, ρh, σ, α, θ, ve e. Algoritma aşağıdaki şekilde gösterilmiştir [32].

1. $\rho = \rho_0$ eşitle.
2. Durum sağlanana kadar 3. adımdan 7. adıma kadar sürekli tekrar et.
3. Tüm F_2 düğümlerini işlenmemiş olarak belirle.
4. S veri kümesindeki her giriş vektörü için 4.a - 4.f arasındaki adımları uygula.

   a) Tüm F_1 düğümleri v_i ve F_2 düğümleri v_j için h_{ij} yi hesapla. Eğer tüm F_2 düğümleri işlenmemişse adım 4.c ye git.
   b) Tüm işlenmiş F_2 düğümleri için T_j değerini hesapla.
   c) Kazanan F_2 düğümünü Vj seç. Hiç F_2 düğümü seçilememişse giriş verisini aykırı gözlem 0 'a koy ve adım 4'ten devam et.
   d) Kazanan bir işlenmiş düğüm ise r_j'yi hesapla, değilse 4.f adımına git.
   e) Eğer r_j>=ρ ise adım 4.f ye git, aksi halde kazanan V_j yi sıfırla ve adım 4.c 'ye geri dön.
   f) Kazanan V_j' yi işlenmiş yap ve dip-yukarı ve üst-aşağı kalınlıklarını kazanan düğüm V_j için güncelle.

5. Stabil küme oluşana kadar adım 4' ü N defa tekrarla. (örneğin N ve N-1 çıkış kümeleri arasındaki fark yeteri kadar küçük olunca)
6. F_2 katmanındaki her C_j kümesi için birleştirilmiş boyut seti D_j' yi hesapla. Ardından S= C_j ve $\rho = \rho + \rho_h$ yaptıktan sonra Adım 2'ye geri git.
7. Aykırı gözlem O için S=0 yap ve Adım 2'ye geri git.

2.4.8. Trees.J48

Aydoğan, (2008) aktardığına göre J48 algoritması C4.5 karar ağacı algoritmasının WEKA (Waikato Environment for Knowledge Analysis) açık kaynak kodlu paket programına uyarlanmış versiyonudur. J48 algoritması; aynı veri seti üzerinden WEKA paket programında Naïve Bayes, Lojistik Regresyon, ID3, JRIP, PART ve Sinir Ağları gibi bilinen sınıflandırma algoritmalarına göre doğruluğu en yüksek sınıflandırma algoritmasıdır [35].

Bu yöntem, öğrenme aşamasında, normalize edilmiş, bilgi kazancı kavramını kullanır. Bu kavram, her bir ağaç, düğümüne giren örüntü kümesini en verimli biçimde bölebilecek olan özniteliği seçmek için kullanılan bir bölme kıstasıdır. Oluşturulacak tüm düğümler için, o düğümde en yüksek kazancı veren öznitelik kullanılarak bir eşik değeri (ξ) bulunur. Bu eşik değeriyle, ilgili düğümü dallara ayıracak kural olan $a_k \leq \xi$ eşitsizliği

oluşturulur. Bu eşitsizlikte, a_k, ilgili düğümünden geçen örüntüleri sınıflandırmak için kullanılacak olan özniteliktir. Eşitsizliğin doğru veya yanlış olma durumuna göre her bir düğüm iki dala ayrılır. Bu işlem, yaprak düğümlerine ulaşılana kadar devam eder. Her bir yaprak düğümü farklı bir sınıfı temsil etmektedir [36].

Altıkardeş, (2012) aktardığına göre J48, enformasyona dayalı, verilerden ilgili özellikleri seçmek için otomatik işlem yeteneğine sahip, Naïve Bayes, ID3, Lojistik Regresyon gibi algoritmalarına göre sınıflandırması en yüksek algoritmadır. Enformasyon kazancının en iyi olduğu noktadan örnekleri bölen yinelemeli algoritmadır. IF-THEN kurallarına dayalı bir karar ağacı ve üyelik fonksiyon kümeleri çıktısı verir. Ağaç yapısı, denekleri bölme ve ağaç en iyi kök değişkeninin seçilmesi süreci ile başlayıp yukarıdan aşağıya doğru inşası gerçekleştirilmektedir. J48 anlamlı olmayan diğer bir deyişle zayıf dalları kesmek için etkin bir budama işlemi yapabilmektedir. Bunun nedenlerinden biri, karar ağaçlarının amacının veri keşfetmek değil, veriler üzerinde basit bir sınıflandırma modeli oluşturmak olmasındandır [37].

C4.5 algoritmasının normal karar ağacı oluşturma algoritmalarından farkı [37];

- Bütün öznitelikler için üzerinde çalışılan veri kümesine ait entropi hesaplanır.
- En küçük entropiye sahip öznitelik bulunduğunda veri kümesinin bölünmesi gerçekleştirilir.
- Bu özniteliği içerek yeni bir karar ağacı oluşturulur.
- Bu işlemler bütün öznitelikler için tekrarlanır.
- Yüksek entropi değerine sahip öznitelik sınıflandırmanın mükemmelleştirilmesinde kullanılır. Eğer hesaplanan Entropi **H(S)=0** olursa **S** kümesi mükemmel sınıflandırılmış demektir. Entropi şu formülle hesaplanır [38].

$$H(S) = -\sum_{x \in X} p(x) \log_2 p(x) \qquad (2\text{-}2)$$

1. S=Entropi hesaplamak için kullanılacak her bir sınıf
2. X=S veri kümesi içindeki her bir sınıf
3. $p(x)\frac{\text{X sınıfı içindeki eleman sayısı}}{\text{S sınıfı içindeki eleman sayısı}}$

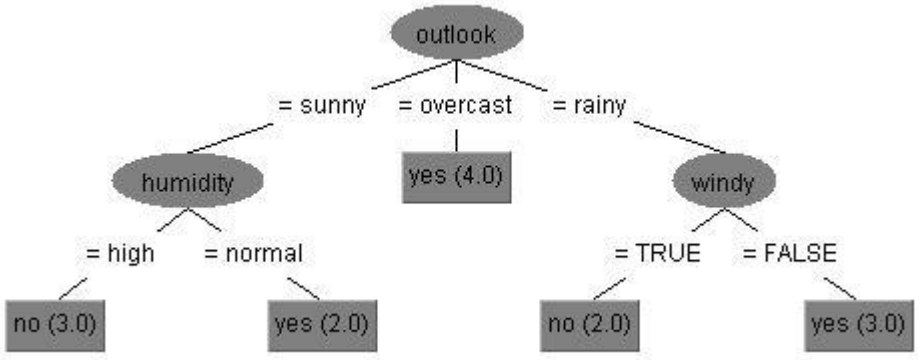
Bilgi Kazancı(Information Gain), entropiden farklı olarak her bir öznitelik için hesaplanmaktadır. Yani bir A özniteliği için bilgi kazancı, o özelliğin bağlı bulunduğu bütün veri kümesine ait entropinin, sadece A özniteliğine ait ağacın entropisi arasındaki

farka eşittir. Ağacın bölünmesi sonrasında elde ettiğimiz kazanımı sayısal olarak ifade etmektedir. En büyük bilgi kazancına sahip öznitelik veri kümesine ait ağacın bölünmesinde kullanılır. Bilgi Kazancı şu formülle hesaplanır [38]

$$IG(A) = H(S) - \sum_{t \in T} p(t)H(t) \quad (2\text{-}3)$$

1. H(S)=S kümesinin entropisi
2. T=S veri kümesinin ayrılması ile oluşan A özniteliğine ait alt küme
3. $\mathrm{p(t)}\,\frac{\text{t sınıfı içindeki eleman sayısı}}{\text{S sınıfı içindeki eleman sayısı}}$
4. H(t)= t altkümesinin entropisi

Weka 3.7 programının kurulumunda varsayılan olarak yüklenen ve bilgisayarımızda "C:\Program Files\Weka-3-7\data\" yolunda bulunan weather.nominal.arff isimli veri kümesinde J48 algoritması kullanılarak elde edilen karar ağacı görünümü aşağıdaki gibidir.



Şekil 2.5 J48 Karar Ağacı

2.4.9. Trees.LMT

Ardıl, (2009) aktardığına göre LMT (Logistik Model Tree) Lojistik Model Ağacı algoritması, lojistik regresyon ve karar ağacı öğrenmesi algoritmalarını birleştiren denetimli bir sınıflandırma modelidir. Bu ağaçlar yapraklarında lojistik azalma fonksiyonları olan sınıflandırma ağaçlarıdır [37]. Landwehr, (2003) aktardığına göre sıradan karar ağaçları yaprakları parçalı bir sabit modeli oluştururken, lojistik modeli

ağacı yaprakları parçalı bir doğrusal regresyon sağlayan, doğrusal regresyon modeline sahip bir karar ağacıdır [39].

LMT de yapraklar, sınıf etiket yerine ilişkili mantık regresyon fonksiyonlarına sahiptir. LMT yaprak düğümleri de lojistik regresyon fonksiyonları ile standart bir karar ağacı yapısıyla örnek bir ağacı oluşturur [40].

Algoritmanın çalışma adımları aşağıdaki gibidir [41];

1.Adım: Başlangış ağacı yetiştirme – Başlangıç doğrusal regresyon modeli, LogitBoost algoritmasının veri kümesinin tamamında sabit sayıda tekrarlanarak kullanılması ile ana düğüm için oluşturulur.

2.Adım: Ayırma ve Durma – LMT algoritmasındaki ayırma kriterleri C4.5 algoritması ile aynı şekilde kullanılır.

3.Adım: Ağırlık oluşumu – Logistic regression modelleri LogitBoost algoritması kullanılarak veri kümesinin altkümelerine karşılık gelen alt düğümleri inşa edilir. Başlangıç ağırlığı ve olasılık tahmini üst düğümden öğrenilir. Ayırma ve model oluşturma, düğümde 15 örnek kalana ve faydalı ayırma bulunana kadar devam eder.

4.Adım: Ağaç budama - Budama yöntemi, budama kararlar modeli karmaşıklığı eğitim hatası ve ceza süresinin bir kombinasyonunu kullanır [41].

2.4.10. Trees.RandomForest

Breiman, L. (2001) aktardığına göre Random Forest karar ağaçlarında kullanılan bir sınıflandırma algoritması olup Leo Breiman ve Adele Cutler tarafından geliştirilen ve içerisinde oylama metodunu barındıran bir sınıflama yöntemidir. Birçok karar ağacının biraya gelmesiyle oluşur ve bireysel ağaçlar tarafından oylanarak kazanan sınıf belirlenir. Karar ağaçları, birbirinden bağımsızdır ve veri setinden bootstrap tekniği ile çekilen örneklerden oluşturulur [29.

Random Forest'ta(RF) kullanılacak veri seti için eğitim ve test verisi önceden belirlenmemişse, veri setindeki sınıf oranları dikkate alınarak tüm veri setinin 2/3'si eğitim ve 1/3'i test verisi olarak kullanılır. Karar ormanını oluşturacak K tane karar ağacı için, gene K adet bootstrap tekniği kullanılarak örneklem oluşturulur ve her bir örneklem için eğitim ve test verisi ayrılır. Tüm ağaçlar ayrılan test verisi ile test edilerek hata oranı hesaplanır ve ardından bu hata oranlarının ortalaması alınarak karar ormanının test hatası hesaplanır. Hesaplanan test hata oranına göre tüm ağaçlara bir ağırlık verilir. Hata oranı

ve ağaca verilen ağırlık değeri ters orantılıdır. Hata oranı en yüksek olan karar ağacı en düşük ağırlığı, hata oranı en düşük olan karar ağacı ise en yüksek ağırlığı alır. Belirlenen ağırlıklara göre tüm ağaçlar sınıflandırma işlemi için bir oylama işleminden geçirilir. En yüksek oyu alan ağaç sınıf tahmini olarak belirlenmiş olur [42].

RF modeli 2 parametre üzerine kuruludur. Bu parametreler; oluşturulacak olan ağaç sayısı (B) ve her düğüm ayrımında rasgele seçilecek olan tahminci sayısıdır (m). Her karar ağacı oluşturulurken, orjinal veri setindeki gözlem sayısı (n) ile aynı ölçüde olacak şekilde bootstrap yöntemi ile örneklem oluşturulur. Bu örneklemin 2/3'ü ağacı oluşturmak için kullanılan eğitim veri seti (inBag) ve geriye kalan 1/3'ü ise kurulan modelin iç hata oranını test etmek için test veri seti (out of bag veya OOB) olacak şekilde ikiye ayrılır [43].

RF algoritması aşağıdaki şekilde kurulur:

1) Bootstrap yöntemi ile n hacimli veri seti seçilir. Bu veri seti, eğitim veri seti (inBag) ve test veri seti (OOB) olarak ikiye ayrılır.

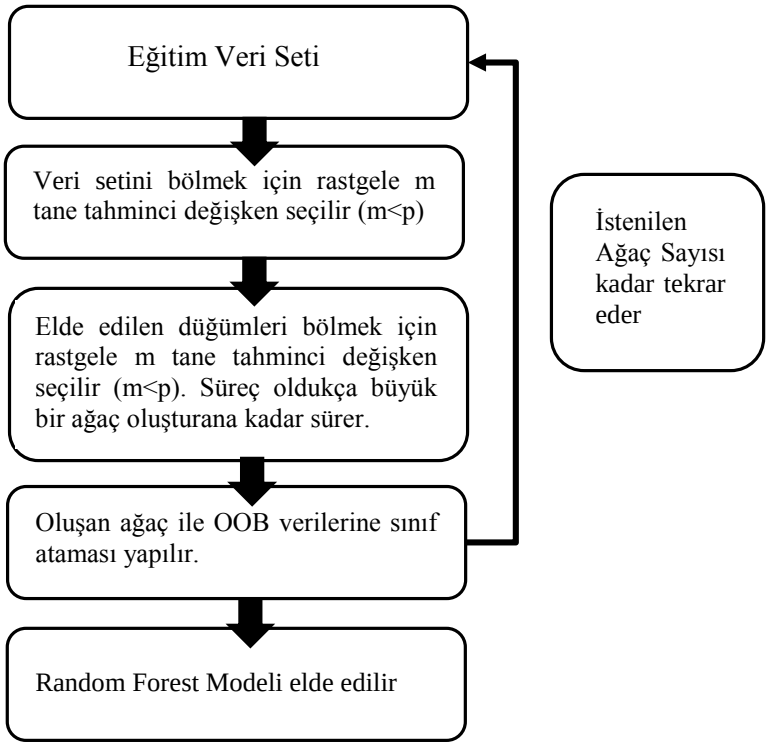
2) Eğitim veri seti (inBag) ile en büyük genişlikte bir karar ağacı (CART) oluşturulur ve elde edilen bu karar ağacı budanmaz. Bu ağaç oluşturulurken her düğümün bölünmesinde toplam p tane tahminci değişkenden m tanesi rasgele seçilir. Burada m<p koşulu sağlanmalıdır. Çünkü ağacın aşırı büyümesi ve aşırı uyum gözlenmesi istenmemektedir. Seçilen bu m tane tahminciden bilgi kazancı en yüksek olan ile dallara ayrılma gerçekleşir. Belirlenen bu değişkenin hangi değerine göre ayrımın olacağına Gini indeksi ile karar verilir. Bu işlem her düğüm için yeni oluşturulacak dal kalmayıncaya kadar tekrar edilir.

3) Her yaprak düğüme bir sınıf atanır. Daha sonra test veri seti (OOB) ağacın en tepesinden bırakılır ve bu veri setinde yer alan her gözleme atanan sınıf kaydedilir.

4) 1.'den 3. adıma kadar tüm aşamalar B defa tekrar edilir.

5) Ağaç oluşturulurken kullanılmayan gözlemler (OOB) ile bir değerlendirme yapılır. İncelenen bir gözlemin hangi kategorilerde kaç defa sınıflandırıldığı sayılır.

6) Her gözleme, ağaç setleri üzerinden belirlenen bir oy çoğunluğu ile sınıf ataması yapılır. Örneğin 2 kategoriye sahip bir sınıflandırma modelinde, bir gözlem tüm ağaçlar üzerinden en az %51 oy çoğunluğunu aldığı sınıfın etiketini taşır ve bu sınıf onun tahmin edilmiş sınıf değeri olur [43].



Şekil 2.6 Random Forest Model oluşturma Algoritması [43]

Random Forest Algoritmasının Avantajları:

Random Forest sınıflama yönteminde her ağaçta verilerin ve değişkenlerin farklı olmasından dolayı aşırı uyum sorunu oluşmaz. Eksik verinin olduğu durumlarda ve çok büyük veri setlerinde rahatlıkla kullanılabilir, büyük veri kümelerinde yüksek başarımlı sonuçlar üretebilir. Ağaç sayısında herhangi bir kısıt yoktur tamamıyla araştırmacının isteğine bırakılmıştır. Bağımsız değişkenlere ilişkin sınırlama yapmaz (kesikli, sayısal, kategorik). İkili (binary) ya da çoklu (multiple) sınıflamada kullanılabilir. Her ağaçta verilerin ve değişkenlerin farklı olmasından dolayı RF, karar ağaçlarındaki gibi budamaya ihtiyaç duymaz. Değişkenleri önem sırasına göre otomatik olarak algoritma içinde sıralayabilir. RF sınıflama yönteminde analistin sadece 2 parametreyi belirlemeye ihtiyacı vardır (1.ağaç sayısı, 2.her düğümde rastgele seçilecek olan değişken sayısı), kullanımı kolay ve öğrenim süreci hızlıdır [29].

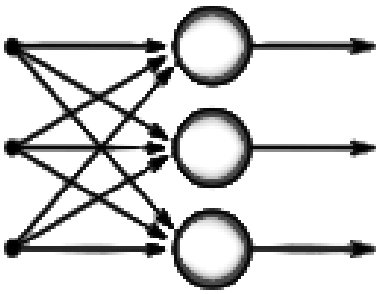
Random Forest Algoritmasının dezavantajları:

Random Forest yöntemi, sınıflamanın doğruluğuna ilişkin bir güven aralığı vermez, çünkü yapısında çok fazla ağaç bulundurmaktadır. Bu sebeple ağaçlar hem şekilsel olarak görülemez hem de belli bir güven aralığı hesaplanmasını engeller. Ancak

Random Forest algoritmasında var olan "bootstrap" tekniği ile zaten yapılan sınıflama genellenmektedir. Bu nedenle güven aralığına ihtiyaç duyulmaz [29].

2.4.11. Functions.VotedPerceptron (Seçimli Algılayıcı)

Seçimli Algılayıcılar yapay sinir ağlarında kullanılan yöntemin ağırlıklandırma sürecinde bir değişiklik yapan ve çekirdek hilesi kullanılarak uyarlanmış bir yapay sinir ağları uygulamasıdır. [44,45] Orijinal yöntemde $\omega.x+b>0$ formülü kullanılmaktadır. ω ağırlıklandırma vektörü, x özellik vektörü b eğilim olmaktadır. Eşitlikte sonuç doğru çıkarsa çıktı 1, yanlışsa çıktı 0 olur. Orijinal algılayıcı eğitim sürecinde özellik vektörü ardışık olarak işleme sokularak bir sınıflandırma eğitimi gerçekleştirilir. Belli bir eşik değerin üzerinde ağın çıkışında hata alındığında ağırlıklar güncellenmektedir. Yapay sinir ağları birden çok algılayıcıyı iç içe kullanır. Eğer veri doğrusal bir şekilde ayrılmaya uygunsa yapay sinir ağları orijinal algoritması başarılı bir sonuca ulaşmaktadır [44,45].



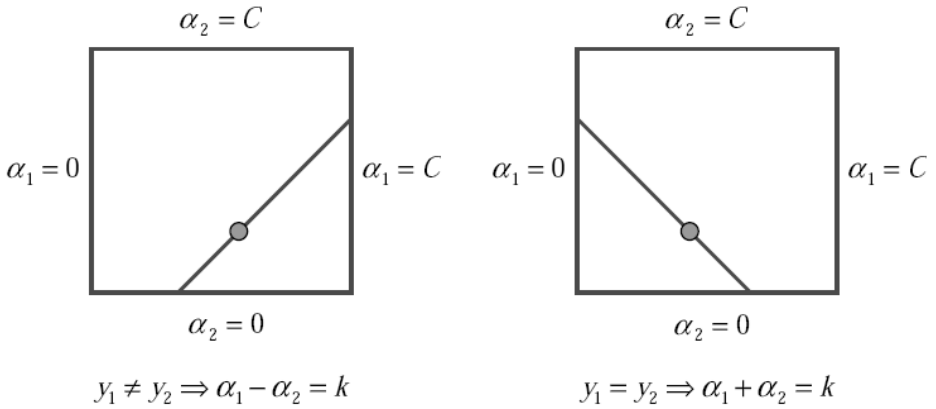
Şekil 2.7 Yapay Sinir Ağları [44]

Verinin karakteri doğrusal değilse seçimli algılayıcılar kullanılabilir. Seçimli algılayıcı algoritması orijinal algoritmanın sürecinde ortaya çıkan tüm ağırlıklandırmaları da hesaba katarak, her ağırlıklandırma vektörünün doğru sınıflandırdığı örnek sayılarını kullanarak tahminde bulunmaya çalışır [44,45].

2.4.12. Functions.SMO (Ardışık Minimal Optimizasyon)

Ardışık Minimal Optimizasyon (SMO- Sequential Minimal Optimization) algoritması Destek Vektör Metodu(SVM) ve İkinci Derece Programlama(QP) problemini ekstra matris alanı ve nümerik ikinci derece programlama optimizasyon adımları kullanmaksızın hızlıca çözebilen basit bir algoritmadır. SMO yakınsamayı koruyarak Osuna teoremi yoluyla tüm ikinci derece programlama problemlerini alt problemlere ayrıştırmaktadır. SMO her adımda en küçük optimizasyon problemini

çözmeyi seçmektedir. Lagrange çarpanları lineer eşitlik kurallarına uymak zorunda olduğundan standart SVM QP problemi için en küçük optimizasyon problemi iki Lagrange çarpanı içermektedir. Her adımda SMO iki Lagrange çarpanını birlikte optimize ederek bu çarpanlar için en iyi değeri bulmaktadır ve SVM' yi yeni en iyi değerleri yansıtarak güncellemektedir.



Şekil 2.8 İki Lagrange çarpanı problemin tüm kısıtlamalarını karşılamak zorundadır. Eşitsizlik kısıtlaması Lagrange çarpanlarının kutu içinde kalmalarına neden olur. Lineer eşitlik kısıtlamaları çarpanların diagonal çizgi üzerinde kalmalarını sağlar. Buna karşın SMO'nun bir adımı bir diagonal çizgi parçasındaki en iyi hedef fonksiyonunu bulmak zorundadır.

SMO'nun avantajı iki Lagrange çarpanının çözümünü analitik olarak yapabilmesidir. Nümerik QP optimizasyonundan tamamen kaçınmaktadır. Algoritmanın iç döngüsü QP kütüphane rutinleri yerine kısa bir C koduyla açıklanabilir. Her ne kadar daha fazla optimizasyon alt problemleri çözülmesini gerektirse de alt problem çözümü çok hızlıdır ve tüm QP problemi hızlıca çözülmektedir.

Ayrıca SMO algoritmasının ekstra matris alanı gerektirmemesi de çok büyük SVM öğrenme problemlerine göre önemli bir bellek avantajı sağlar. SMO içinde hiç matris algoritması kullanılmaması nümerik duyarlılık problemlerinden daha az duyarlılık elde edilmektedir [46].

2.5. Sınıflandırma Algoritmalarının Karşılaştırılmasında Kullanılan Kriterler

Çalışmada kullanılan veri kümeleri, başarının göstergesi olarak kullanılan sontest özniteliği verisine uygun tüm sınıflandırma algoritmaları ile test edilmiştir. Çalışmada kullanılan tüm sınıflandırma algoritmaları varsayılan değerleri değiştirilmeden ve test seçeneği olarak 10 kat çaprazlama (10 fold cross-validation) tekniği ile kullanılmıştır. Bu

yöntemde veri kümesi 10 eşit parçaya bölünür. 10 eşit parçadan 1 tanesi test için, geriye kalan 9 tanesi eğitim için kullanılır [47].

Bir sınıflandırıcının öğrenme sürecinin sonucunda ne kadar başarılı olduğuna karar verilmesi için çeşitli ölçütlere gereksinim vardır. En temel başarım ölçütleri; doğruluk oranı, kesinlik, duyarlılık ve f ölçütüdür. Ayrıca modelin başarısı ölçülürken, yukarıda verilen ölçütlerin dışında, kappa istatistiği de kullanılabilir [47].

Bu çalışmaya konu olan öğrenme yönteminin başarımını ölçmek için kullanılan ölçütler şunlardır: Doğru tahmin oranı, F-Ölçütü değeri, Kappa İstatistiği, Ortalama Karesel Hatanın Karekökü. Bu ölçütler aşağıda açıklanmaktadır.

2.5.1. Karışıklık Matrisi

Modelin başarısı, doğru sınıfa atanan örnek sayısı ve yanlış sınıfa atanan örnek sayısı nicelikleriyle alakalıdır. Test sonucunda ulaşılan sonuçların başarım bilgileri karışıklık matrisi ile ifade edilebilir. Karışıklık matrisinde satırlar test kümesindeki örneklere ait gerçek sayıları, kolonlar ise modelin tahminini ifade eder [48].

İki sınıfa ayrılmış bir veri kümesi içerisinde yapılan veri ayrıştırma işlemi sonucunda bu matris içerisinde dört durum oluşmaktadır.

Tablo 2.4 Karışıklık Matrisi

		Tahmin Edilen Sınıf	
		Sınıf=Başarılı	Sınıf= Başarısız
Gerçek Sınıf	Sınıf=Başarılı	A	B
	Sınıf=Başarısız	C	D

A: TP (True Pozitif)

B: FN (False Negatif)

C: FP (False Pozitif)

D: TN (True Negatif)

A-TP, Gerçekte başarılı olup, sınıflandırıcı tarafından başarılı sınıfına atananların sayısını gösterir.

B-FN, Gerçekte başarılı olup, sınıflandırıcı tarafından başarısız sınıfına atananların sayısını gösterir.

C-FP, Gerçekte başarısız olup sınıflandırıcı tarafından başarılı sınıfına atananların sayısını gösterir.

D-TN, Gerçekte başarısız olup, sınıflandırıcı tarafından başarısız olarak atananların sayısını gösterir.

2.5.2. Kappa İstatistiği

Witten, (2005) aktardığına göre Kappa İstatistiği (KS) bir veri kümesindeki tahmin edilen ve gözlenen sınıflandırmalar arasındaki uyumu nicel olarak ifade etmek için kullanılan bir ölçüttür [14].

Landis, J., vd., (1977) aktardığına göre Kappa katsayısı -1 ile +1 arasında değişir. Tam uyum söz konusu olduğunda K=1 olur. Gözlenen uyumun şansa bağlı uyuma eşit ya da ondan büyük olması durumunda K≥0 iken, gözlenen uyumun şansa bağlı uyumundan küçük olması durumunda K<0 olur. Kappa katsayısının yorumlanabilir aralığı 0 ile +1 arasında olup, negatif (K<0) değerlerinin güvenirlik açısından bir anlamı yoktur. 0,4 üzerinde bir kappa değeri makul bir anlaşmayı ifade eder. Kappa istatistik değerinin 0,6 ile 0,8 arasında olması önemli derecede uyum olduğunu, 0,8 ile 1 arasında olması ise neredeyse mükemmel bir uyumun olduğunu göstermektedir. Aşağıda Landis vd.. sunduğu kappa tablosu yer almaktadır [49].

Tablo 2.5 Kappa Değerleri

Kappa Değeri	Yorum
< 0	Hiç uyuşma olmaması
0.0 — 0.20	Önemsiz uyuşma olması
0.21 — 0.40	Orta derecede uyuşma olması
0.41 — 0.60	Ekseriyetle uyuşma olması
0.61 — 0.80	Önemli derecede uyuşma olması
0.81 — 1.00	Neredeyse mükemmel uyuşma olması

Kappa istatistiği değeri aşağıdaki formülle hesaplanmaktadır.

$$\boldsymbol{K} = \frac{(P_a - P_c)}{(1 - P_{c)}} \tag{2-6}$$

Kılıçaslan vd., (2009) aktardığına göre P(a) sınıflandırıcının doğruluğunu göstermektedir ve P(c) aynı veri kümesi üzerinde rastgele bir biçimde tahmin yapan sınıflandırıcının elde etmiş olduğu beklenilen doğruluktur [14].

2.5.3. Model Başarım Ölçütleri

2.5.3.1. Doğru tahmin oranı

Doğru tahmin oranı, model başarımının ölçülmesinde kullanılan en popüler ve basit yöntemdir. Doğru tahmin oranı sınıflandırıcı algoritmaların veri kümesi içerisindeki gözlemleri hem eğitim hem de test verisi olarak kullandığı ve sonuç olarak doğru sınıflandırma oranını veren değerdir [24].

Doğru sınıflandırılmış örnek sayısının (TP +TN), toplam örnek sayısına (TP+TN+FP+FN) oranıdır. Hata oranı ise bu değerin 1'e tamlayanıdır. Diğer bir ifadeyle yanlış sınıflandırılmış örnek sayısının (FP+FN), toplam örnek sayısına (TP+TN+FP+FN) oranıdır [49].

$$\boldsymbol{Doğruluk} = \frac{(TP+TN)}{(TP+FP+FN+TN)} = \frac{Algoritmanın\ Yaptığı\ Doğru\ Tahmin\ Sayısı\ Toplamı}{Örnek\ Sayısı} \tag{2-4}$$

$$\boldsymbol{Hata} = \frac{(FP+FN)}{(TP+FP+FN+TN)} = \frac{Algoritmanın\ Yaptığı\ \text{Yanlış}\ Tahmin\ Sayısı\ Toplamı}{Örnek\ Sayısı} \tag{2-5}$$

2.5.3.2. F-Ölçütü

F-Ölçütü (F-Measure) değeri precision ve recall değerlerinin harmonik ortalaması olarak ifade edilir. F-ölçütü özellikle eğitim verilerinin hazırlanış sürecinde sınıflandırıcının performansını arttırmak için önemli bir ölçüt olmakla birlikte sınıflandırıcıların performans analizinde f-ölçütü ve veri büyüklüğü arasındaki ilişki ortaya konularak f-ölçütü ve veri büyüklüğü arasında çizilen öğrenme eğrisinden bir mutluluk grafiği (Happy Graph) elde edilmeye çalışılmıştır [50,14]

Bunun yanı sıra geliştirilen uygulama da F-ölçütü sınıfların tanı için yeterli olup olmadığını tespit etmek amacıyla kullanılmıştır. Kabul edilebilir F-ölçütü değeri genellikle minimum 0,5 olarak alınır. Bu tez çalışmasında da bu değer 0,5 olarak alınmıştır [14].

Table 2.6 Karışıklık Matrisi

		Tahmin Edilen Sınıf	
		Sınıf=Başarılı	Sınıf= Başarısız
Gerçek Sınıf	Sınıf=Başarılı	TP	FN
	Sınıf=Başarısız	FP	TN

Precision: Kesinlik, sınıfı 1 olarak tahmin edilmiş True Pozitif (TP) örnek sayısının, sınıfı 1 olarak tahmin edilmiş tüm örnek sayısına (TP+FP) oranıdır.

$$\boldsymbol{Precision} = \frac{TP}{TP+FP} \quad (2\text{-}7)$$

Recall: Doğru sınıflandırılmış True Pozitif (TP) örnek sayısının gerçek toplam pozitif (TP+FN) örnek sayısına oranıdır.

$$\boldsymbol{Recall} = \frac{TP}{TP+FN} \quad (2\text{-}8)$$

F-Measure değeri aşağıdaki formülle hesaplanır.

$$\boldsymbol{F-Measure} = \frac{2*\,Precision*\,Recall}{Precision+\,Recall} \quad (2\text{-}9)$$

2.5.4. Ortalama Karesel Hatanın Karekökü (RMSE)

Challagulla, (2005) aktardığına göre RMSE model veya tahmin edici tarafından tahmini yapılan değerler ve modellenen veya tahmini yapılandan elde edilen gerçek değerler arasındaki farkın ölçüsü olarak sıklıkla kullanılır [51].

Witten vd., (2005) aktardığna göre Ortalama Karesel Hatanın Karekökü (RMSE) Ortalama Karesel Hata değerinin (MSE) karekökünün alınmasıyla hesaplanır [14]. Ortalama karesel hata (MSE), sayısal kestirimler için en çok kullanılan başarı ölçülerinden biridir. Bu değer, her hesaplanan değer ve onun karşılık gelen doğru değer arasındaki farkın karelerinin ortalaması alınarak hesaplanır [14]. MSE ve RMSE'nin hesaplanmaları aşağıda gösterilmektedir. Buna göre p, yapılan tahmini değerleri; a ise gerçek değerleri ifade etmektedir.

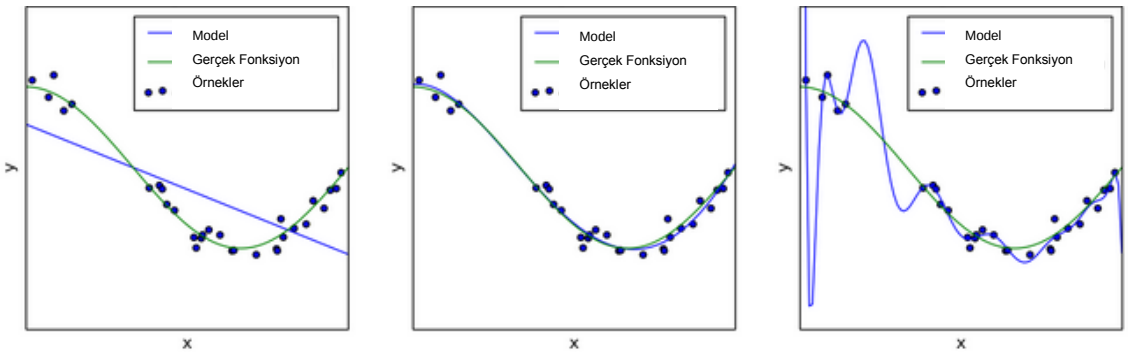
$$\boldsymbol{MSE} = \frac{(P_1-a_1)+\cdots+(P_n-a_n)}{n} \quad (2\text{-}10)$$

$$\boldsymbol{RMSE} = \sqrt{\frac{(P_1 - a_1) + \cdots + (P_n - a_n)}{n}} \quad (2\text{-}11)$$

2.6. Yetersiz Uyum (underfitting) ve Aşırı Uyum (overfitting)

Bir makine öğrenmesi algoritması ilgili eğitim verisi için sınıflama performansını maksimum yapmaya çalışmaktadır. Eğitim verisinin sınıflama performansı maksimum olduğunda test verisi ile yapılan sınıflamanın doğruluğu çok daha yüksek olmaktadır. Ancak öğrenme sırasında algoritma eğitim setine çok uyarsa aşırı uyum(overfitting) olarak adlandırılan durum ortaya çıkmaktadır. Böyle bir durumda algoritmanın sınıflama performansı düşük olur ve genelleme yeteneği azalır. Genelleme yeteneği ile eğitim verisine uyum arasında ters orantı vardır [52].

Eğitim verisi kullanılarak teorik olarak sıfır hata oranına sahip karar ağacı oluşturmak mümkündür. Ama oluşturulan model için aşırı öğrenme (overfitting) durumu ortaya çıkabilir. Aşırı öğrenme durumunda oluşturulan model öğrenme verisi için çok başarılı sınıflama sağlarken, yeni veriler için doğruluk oranı çok düşük olabilir. Aşırı öğrenme, karar ağaçları çok fazla detayı karakterize ettiğinde veya öğrenme verisinde çok fazla gürültülü veri olduğunda ortaya çıkmaktadır [53].



Şekil 2.9 3 Farklı Model Kullanılarak Oluşturulmuş Sınıflandırma Örneği [54]

Şekilde çeşitli öğrenme tipleri yer almaktadır. Yaklaşılmak istenen fonksiyon bir kosinüs fonksiyonunun parçasıdır ve yeşil renkle çizilmiştir. Noktalar gerçek fonksiyonun örnekleri, mavi çizgiler ise farklı modellerin yaklaşımlarıdır. Birinci resimde görülen model birinci derece bir polinom kullanılarak oluşturulan bir lineer fonksiyondur ve eğitim örneğine yeterli miktarda uyum göstermemektedir. Bu durum yetersiz uyum (underfitting) olarak adlandırılmaktadır. İkinci resimde dördüncü derece bir polinom

kullanılarak oluşturulan yaklaşım fonksiyonu gerçek fonksiyona nerdeyse mükemmel biçimde uymaktadır. Üçüncü resimde ise çok daha yüksek dereceli bir polinom kullanılarak oluşturulan model görülmektedir. Model eğitim verisinin içerdiği kirlilik ve gürültüyü de öğrenmiştir. Bu nedenle eğitim verisi ile aşırı uyumlu (overfitting) olduğu söylenir [54].

2.7. Makine Öğrenmesi Konulu Literatür Araştırması

Bu bölümde Makine Öğrenmesi yaklaşımına ilişkin kaynaklardan ulaşılabilen bazı çalışmalar özetlenmiştir.

Özarslan ve Barışcı (2014) yaptığı çalışmada, Kırıkkale üniversitesinde ENF-101 kodlu dersi yüz yüze eğitim ile alan grupla, uzaktan eğitim yöntemi ile alan iki grup arasındaki ders performans analizi, derse ait başarı notu, okuduğu bölüm, cinsiyeti vb.. özellikler için Weka programı kullanılarak analiz edilmiştir. Çalışma sonunda eldeki veriler J48 algoritması ile yapılan sınıflandırma sonucunda %82,22 ve Çok Katmanlı Algılayıcı ile yapılan sınıflandırma sonucunda %80,74 başarı yakalanmıştır [55].

Alan (2012) veri madenciliği yöntemiyle Cumhuriyet Üniversitesi Sosyal Bilimler Enstitüsü öğrencilerine ait verileri kullanarak bir uygulama yapmıştır. Lisansüstü öğrencilerine ait verilerden yararlanarak, hem bu verileri en başarılı sınıflandıran algoritma, hem de öğrencilerin programı, cinsiyeti, Sivas ilinden ya da başka bir ilden olması, kadrosunun araştırma görevlisi olup olmaması ve ders döneminin farklı olmasının notlarını etkileyip etkilemediği tespit edilmeye çalışılmıştır. Çalışmanın sonucunda SimpleCART algoritmasının sınıflandırma oranı en yüksek algoritma olduğu, doktora programına kayıtlı öğrencilerin yüksek lisans programına kayıtlı öğrencilere göre daha yüksek başarı notlarına sahip oldukları ancak cinsiyetin, kadronun, ilin ve ders döneminin başarıyı etkilemediği ortaya çıkmıştır [56].

Ekren (2012) Sinop Üniversitesine bağlı akademik birimlere, 2010–2011 yılı içinde kayıt yaptıran öğrencilerin kayıt bilgileri üzerinde, açık kaynak kodlu bir veri madenciliği uygulama aracı olan WEKA yardımıyla bir veri madenciliği uygulaması yapmıştır. Çalışma ile öğrencilerin geldiği iller de katılarak periyodik aralıklarla yapıldığı takdirde söz konusu üniversitenin ve bulunduğu bölgenin kültürel yapısında meydana gelen değişiklikler izlenebilir, üniversitenin öğrenci sayısını ve öğrenci

performansını arttırmak için farklı stratejiler geliştirilebilir olduğu gösterilmeye çalışılmıştır [57].

Güner ve Çomak (2011) tarafından yapılan çalışmada, 2007 yılında Pamukkale Üniversitesi mühendislik fakültesine başlayan 434 öğrencinin üniversiteye giriş sınavı sonuçlarına ait verilerin kullanıldığı araştırma sonuçlarına göre; öğrencilerin matematik, fen bilimleri, Türkçe testlerinin sonuçları ile lise mezuniyet başarı puanlarının, matematik I dersindeki başarılarını tahminde önemli rol oynadığı bulunmuştur. 289 öğrencinin verisi ile makine eğitilmiş, 145 öğrencinin verisi ise test için kullanılmış ve matematik I dersinden geçen öğrencilerin % 86'sı doğru olarak tahmin edilmiştir [8].

Ayık ve Özdemir, Yavuz (2007) tarafından yapılan çalışmada Atatürk Üniversitesi öğrencilerinin mezun oldukları lise türleri ve lise mezuniyet dereceleri ile kazandıkları fakülteler arasındaki ilişki, veri madenciliği teknikleri kullanılarak incelenmiştir. Çalışma sonucunda, lise türünün arzu edilen bir fakültenin kazanılmasında çok büyük öneminin olduğu, yine lise başarısının da aynı derecede önemli olduğu tespit edilmiştir [18]

Makine öğrenmesi algoritmalarının kullanımını kapsayan doktora tezlerinden biri Kartal (2015) tarafından sunulmuş olup, tezde sağlık alanında oluşturulmuş bir veri setinden yararlanılmıştır. Sunulan tezde model performans değerlendirme ölçüsü olarak ortalama doğruluk, F-Ölçüsü ve Tanısal Üstünlük Oranı değerlendirmelerde kullanılmıştır. Tez kapsamında hayati riski tahmin edebilmek için Naïve Bayes Sınıflandırıcı, k-En Yakın Komşu Algoritması, Logistik Regresyon Analizi, ID3 ve C4.5 Karar Ağacı Algoritmaları'ndan yararlanılmış, F-ölçüsü, ortalama doğruluk ve ortalama hata oranları dikkate alındıığında en yüksek performansın Naïve Bayes Algoritmasından elde edildiği belirtilmiştir [58].

Çatak (2014) yaptığı tez kapsamında, veri madenciliğinde sınıflandırma ve regresyon problemlerinin çözümünde oldukça sık kullanılmakta olan makine öğrenmesi algoritması DVM algoritmasının eğitim aşamasının "Eşle/İndirge" yazılım geliştirme tekniği ile dağıtık ve paralel olarak nasıl eğitilebileceği araştırmıştır. Kullanılan veri setlerinde elde edilen deneysel sonuçlar ile dağıtık ve paralel DVM sınıflandırma algoritmasında eşle fonksiyon sayısının artması, algoritmanın eğitimine olanak sağlarken sınıflandırıcı hipotez uzayından seçilen deneysel riski en düşük sınıflandırıcı fonksiyonun belirli bir değerin üzerinde düşmeye başladığının gözlendiği raporlanmıştır. Önerilen

metodun performansı, DVM sınıflandırma algoritmasının kullanılan veri seti ile doğrudan bağım olduğu, kullanılan veri setinde korelasyon durumunun artması veya gürültüye sahip olmasının sınıflandırma performansına olumsuz etkisi olduğu belirtilmiştir [59].

Emir(2013) yaptığı tez çalışmasında hisse senedi piyasalarında öngörü yapmak için örneklerden öğrenme yapabilme ve genelleme yeteneğine sahip makine öğrenmesi yöntemlerinden olan Yapay Sinir Ağları ve Destek Vektör Makineleri İstanbul Menkul Kıymetler Borsası Ulusal 100 Endeksi'nin yön tahmininde gösterdikleri sınıflandırma performanslarını incelemiştir. İncelenen veri kümeleri üzerinde en yüksek performansın her zaman YSA tarafından elde edildiği, YSA yönteminin DVM yöntemine göre daha az tahmin hatası oluşturduğu sonucuna ulaşıldığı vurgulanmıştır. Ayrıca çalışmada sınıflandırıcı performanslarının arttırılabilmesinin, parametrelerin daha hassas ve daha geniş bir aralıkta tanımlanması veya veri kümesindeki değişkenlere ilaveten konuyla ilgili başka değişkenlerin eklenmesi ile mümkün olacağı bildirilmiştir [60].

Aydilek (2013) yaptığı tez çalışmasında veri kümesinde bulanan eksik değerleri hesaplamak için özgün hibrit çalışmalar yapmış ve önerdiği yaklaşımlarla daha iyi ve tutarlı sonuçlar elde ettiğini belirtmiştir. Tam nitelik değerleri içeren veri kümesi oluşturmak için destek vektör regresyonu ve genetik algoritmaları kullanarak bulanık c-ortalamalar kümeleme metoduyla eksik değerleri hesaplatmıştır. Önerilen hibrit yaklaşımın kümelemeye uygun veri kümelerindeki eksik değer probleminin üstesinden gelmede tek başına yeterli bir yaklaşım olduğu sonucuna ulaşmıştır. Karşılaştırılan yapay sinir ağları ve genetik algoritmalar için en iyi sonuçların tam verilerle eğitilmiş yapay sinir ağının eğitim performansına bağlı olduğu bildirilmiştir. Daha tutarlı eğitim verisine sahip veri kümelerinin eksik değerlerinin en yakın k-komşu algoritmasıyla daha hassas ve doğru hesaplanabildiği ortaya koyulmuştur [61].

Koçer (2012) yaptığı tez çalışmasında optimizasyon problemlerinde transfer öğrenme tekniklerinin kullanımını sağlamayı amaçlamıştır. Teoride optimizasyon yöntemleri, optimizasyon problemlerinin tüm değişkenlerinin sabit olduğu varsayımına dayanarak mevcut durum için optimum sonuç aradığı fakat gerçek hayatta problem parametrelerinin değişkenlik gösterdiği belirtilmiştir. Hesaplanmış optimum sonuçlardan faydalanarak yeni optimum sonuçlar kullanılarak genetik transfer öğrenme ile çok daha hızlı bir şekilde sonuç üretildiği bildirilmiştir [4].

Aslan (2010) tarafından yapılan çalışmada e-öğrenme sistemleri için Bayes ağı odaklı ve makine öğrenmesine dayalı bir öğrenci modelleme orta katmanı ortaya konulmuş, e-öğrenme sisteminin benzetimi ve gerçek öğrenciler ile deneysel uygulamaları yapılmıştır. Çalışmada ön-test son-test yarı-deneysel deseni kullanılmış,23'er kişilik deney ve kontrol gruplarının kullanımıyla öğrenci modelleme sistemi tabanlı öğrenme ile geleneksel bilgisayar destekli öğrenme arasındaki akademik başarı artışları değerlendirilmiş, öğrenci modelleme sisteminin ürettiği öğrenci profillerinin başarımı ölçülerek literatürdeki benzer çalışmalarla karşılaştırılmıştır. Bu çalışmada öne sürülen makine öğrenmesine dayalı öğrenci modelleme sisteminin öğrenme biçemlerini hangi seviyede başarımla teşhis ettiği hesaplanmıştır. Uygulamalar sırasında elde edilen bulguların istatistiksel çözümlemelerinin ve basarım hesaplarının yapılması sonucunda, bu çalışmada öne sürülen makine öğrenmesine dayalı öğrenci modelleme sistemiyle geleneksel bilgisayar destekli öğrenme sisteminin öğrenme süreci sonunda akademik başarıda artışa olan etkileri arasında anlamlı bir fark bulunamamıştır [62].

Köktürk yaptığı çalışmada veri madenciliği yöntemlerinden, k-en yakın komşuluk, yapay sinir ağları ve karar ağaçları yöntemlerinin sınıflandırma başarılarının karşılaştırılması amaçlanmıştır. Bu amaçla Bülent Ecevit Üniversitesi Uygulama ve Araştırma Hastanesi Kadın Hastalıkları ve Doğum Polikliniği'ne başvuran erken ve zamanında doğum yapan gebelerden elde edilen veri setine bu üç teknik uygulanarak, sınıflandırma başarıları hesaplanmıştır. Yapılan analizler sonucunda doğru sınıflandırma oranları, k-en yakın komşuluk analizi için % 78.3, yapay sinir ağı tekniği için % 90.8 ve karar ağacı yöntemi için ise % 82.5 olarak bulunmuş ve yapay sinir ağı tekniğinin diğer iki yönteme göre sınıflandırma başarısının daha iyi olduğu görülmüştür [63].

Uçar (2013) tarafından yapılan çalışmada Orta Öğretime Geçiş Sınavı (OGES) puanlarının tahmini için ilköğretim 8. sınıf öğrencilerinden rastgele seçilen 25000 kayıt kullanılmıştır. OGES yerleştirme puanlarının tahmin edilmesi için bir sistem tasarlanmıştır. Çalışmada sınıflandırma ve öngörü konusunda en çok tercih edilen veri madenciliği tekniklerinden yapay sinir ağları, regresyon analizi, C4.5 karar kuralı türetme algoritması ve destek vektör makineleri modelleme yöntemi seçilmiştir. Bu dört yöntemin doğruluk oranları ve performansları karşılaştırılarak, en uygun yöntem bulunmaya çalışılmıştır.

Çalışmada rastgele seçilmiş 5000 kayıt üzerinde SPPS programı kullanılarak çapraz doğrulama (10 fold cross validation) yöntemi ile sonuçlar elde edilmiştir. Sonuca en çok etki eden parametrelerin belirlenmesi için duyarlılık analizi kullanılmıştır. Kullanılan yöntemlerin sınıflandırma başarısında karar ağaçlarının %91,584 lük doğrulukla birinci sırada yer aldığı vurgulanmıştır. Karar ağaçlarını %89,09 ile çok katmanlı yapay sinir ağları, %88,404 ile destek vektör makineleri ve %87,294 ile lojistik regresyon analizinin takip ettiği, Radyal tabanlı fonksiyonun ise %72,35 ile sonuncu sırada yer aldığı belirtilmiştir [52].

2.8. WEKA

Veri Madenciliği uygulamalarını gerçekleştirmek için programlara ihtiyaç duyulur. Bu kapsamda, SPSS Clementine, Excel, SPSS, SAS, Angoss, KXEN, SQL Server, MATLAB ticari ve RapidMiner(YALE), WEKA, R, C4.5, Orange, KNIME açık kaynak olmak üzere birçok program geliştirilmiştir. Bu çalışmada Veri Madenciliği Açık Kaynak Kodlu programlarından WEKA programının 3.7.12 sürümü kullanılmıştır.
Weka, Avustralya'daki Waikato üniversitesi tarafından java tabanlı olarak geliştirilmiş ve dünya üzerinde bir çok kullanıcısı olan açık kaynak kodlu bir veri madenciliği (data mining) programıdır [6].

Weka, tamamen modüler bir tasarıma sahip olup, içerdiği özelliklerle veri kümeleri üzerinde görselleştirme, veri analizi, iş zekâsı uygulamaları, veri madenciliği gibi işlemler yapabilmektedir. Temel olarak aşağıdaki 3 Veri Madenciliği işlemi Weka ile yapılabilir:

- Sınıflandırma (Classification)
- Bölütleme (Clustering)
- İlişkilendirme (Association)

Ayrıca yukarıdaki işlemlere ilave olarak, veri kümeleri üzerinde ön ve son işlemler yapılabilir

- Veri Ön işleme (Data Pre-Processing)
- Görselleme (Visualization)

Weka Kütüphanesinde veri içeren dosyalar üzerinde çalışan çok sayıda hazır fonksiyon bulunmaktadır. Weka yazılımı, kendisine özgü olarak bir. arff uzantısı desteği ile gelmektedir [64]. İngilizce, Attribute Relationship File Format kelimelerinin baş

harflerinden oluşmuştur. ARFF dosya yapısı, Weka'ya özel olarak geliştirilmiştir ve dosya, metin yapısında tutulmaktadır. Weka da kullanılacak bir ARFF formatlı belge aşağıdaki gibi görünmektedir. Burada @Relation olarak belirtilen satır veri kümesinin adını, @Attribute olarak belirtilen satırlar veri kümesinde bulunan her bir örneğin sahip olacağı öznitelikleri, @Data başlığı altında bulunan satırlarda belirlenen özniteliklere sahip olan örnekler yer alır. Ayrıca veri kümesindeki her örneğin her özelliği arasında da virgül ayıracı kullanılmaktadır.

```
@relation weather.symbolic

@attribute outlook {sunny, overcast, rainy}
@attribute temperature {hot, mild, cool}
@attribute humidity {high, normal}
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data

sunny,hot,high,FALSE,no
sunny,hot,high,TRUE,no
overcast,hot,high,FALSE,yes
rainy,mild,high,FALSE,yes
rainy,cool,normal,FALSE,yes
rainy,cool,normal,TRUE,no
```

Şekil 2.10 Arff Dosyası Görünümü



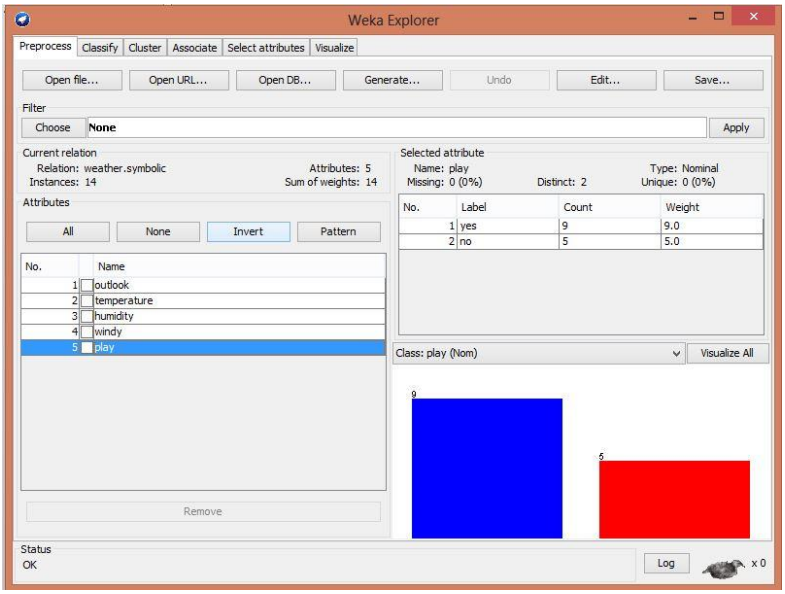
Şekil 2.11 Weka Karşılama Ekranı

WEKA programının karşılama ekranı olarak görüntülenen pencerede Explorer, Experimenter, KnowledgeFlow ve Simple CLI ekranlarına ulaşmak için kullanılan komut düğmeleri bulunmaktadır. WEKA programının genel kullanıcı arayüzü olarak Explorer ekranı geçmektedir.

Weka Explorer penceresinde Preprocess (önişlem), Classify (sınıflama), Cluster (kümeleme), Associate (birliktelik kuralları), Select Attribute (nitelik seçme) ve Visualize (görselleştirme) sekmeleri bulunmaktadır.

Önişlem Ekranı (Preprocess):

Verilerin bulunduğu dosyaların seçildiği ekrandır. WEKA'ya. arff, .csv, .data ve. json formatında dosya yüklemek mümkündür. WEKA programının önişlem ekran görüntüsü ise aşağıda verilmiştir. Bu ekranda eklenen veri kümesi dosyasındaki örnek sayısı, seçilen özniteliğin tipi, seçili özniteliğin dosyadaki minimum, maksimum, ortalama ve standart sapma değerleri ve ilgili özelliğin dağılımı görüntülenmektedir. Bu bölümde classify value olarak anılan bir özellik vardır. Bu özellik veri kümesi dosyasında tanımlanan en son özniteliktir. Önişlem ekranın sağ altında bulunan görüntüleme ekranında, veri kümesinde tanımlanmış öznitelik listesinden herhangi bir öznitelik seçildiğinde ilgili özniteliğin dağılımı gösterilirken classify value 'ya ait renkler ile gösterilir.

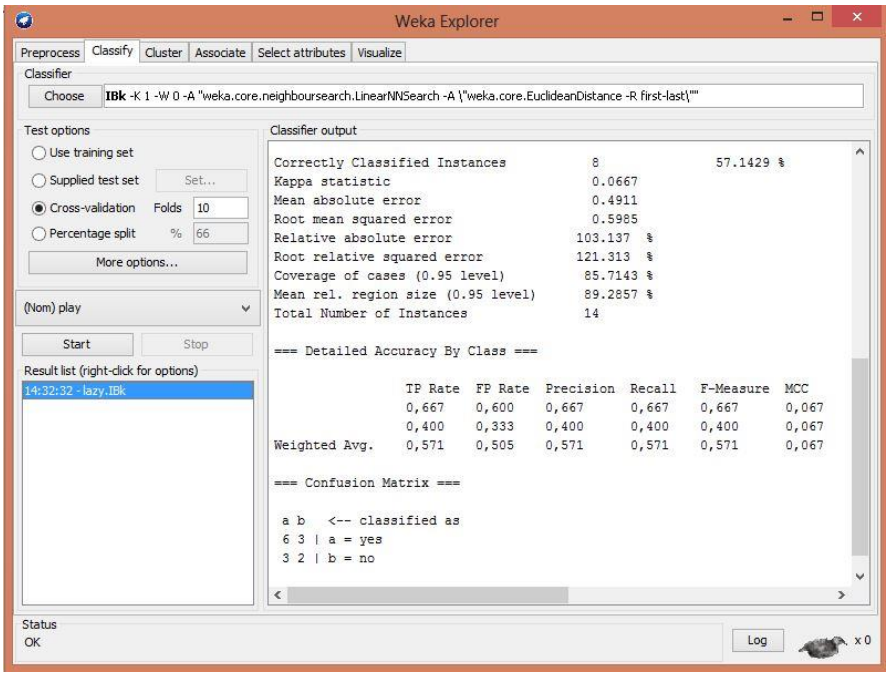


Şekil 2.12 Weka Önişlem Ekranı

Sınıflandırma Ekranı (Classify):

Veri kümesi üzerinde WEKA'nın sınıflandırma algoritmalarının seçilip, çalıştırıldığı ve sonuçlarının görüntülendiği ekrandır. Kurulacak modeli belirlemek için farklı algoritmalar veri kümesi üzerinde çalıştırılarak farklı test biçimleri kullanılarak karşılaştırılmalıdır.

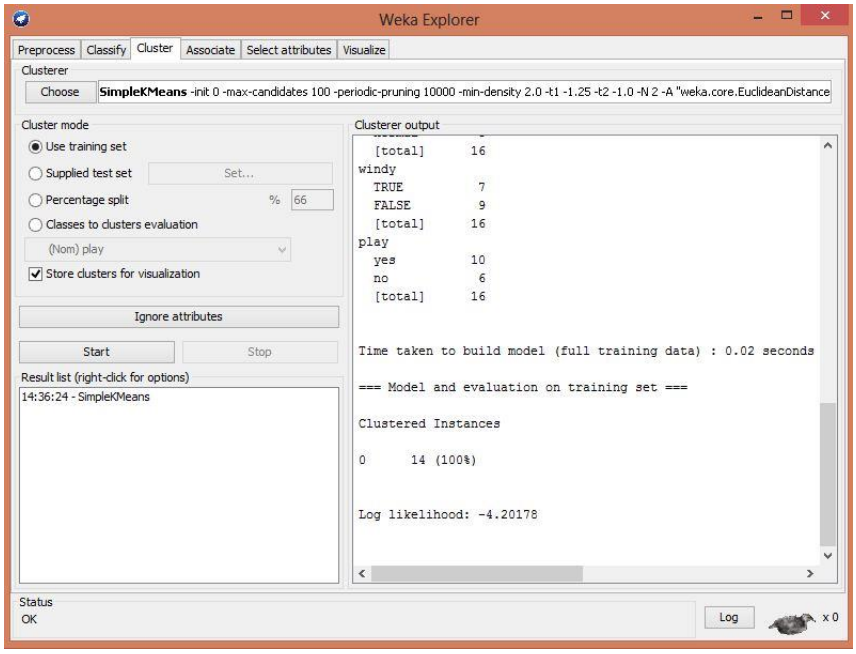
Bu ekranda bulunan **Choose** komut düğmesine basılarak weka'nın sahip olduğu sınıflama algoritmalarının bulunduğu menünün açılması sağlanır. Ayrıca ekrandaki **Test Options** bölümünde yer alan seçenekleri kullanılarak veri kümesinde bulunan örneklerin ne kadarının eğitim ne kadarının test verisi olarak kullanılacağı belirlenmektedir.



Şekil 2.12 Weka Sınıflandırma Ekranı

Kümeleme Ekranı (Cluster):

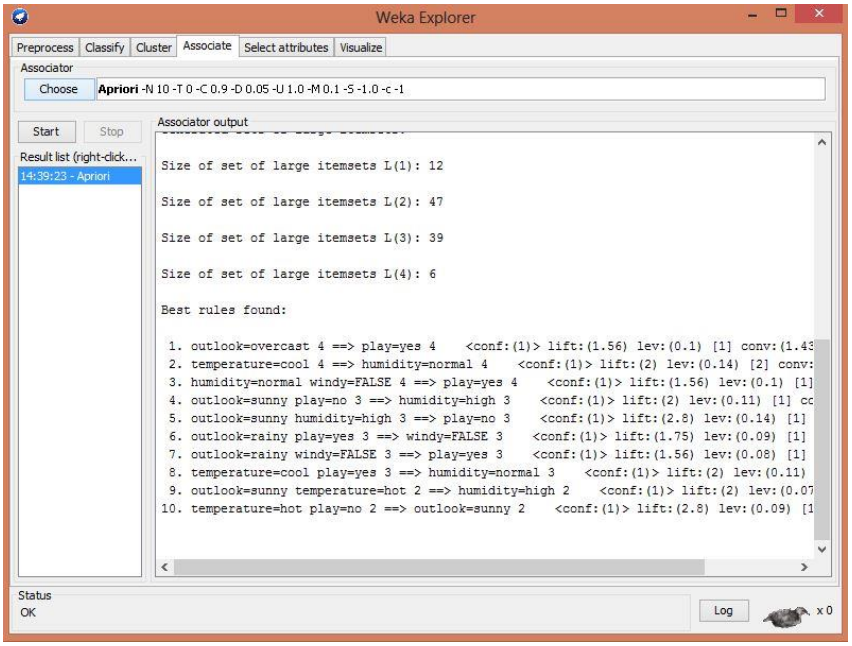
Bu ekranda veriler üzerinde WEKA kümeleme algoritmaları belirlenen kümeleme modları kullanılarak çalıştırılır ve sonuçları gözlemlenir.



Şekil 2.13 Weka Kümeleme Ekranı

Birliktelik Kuralları Ekranı (Associate):

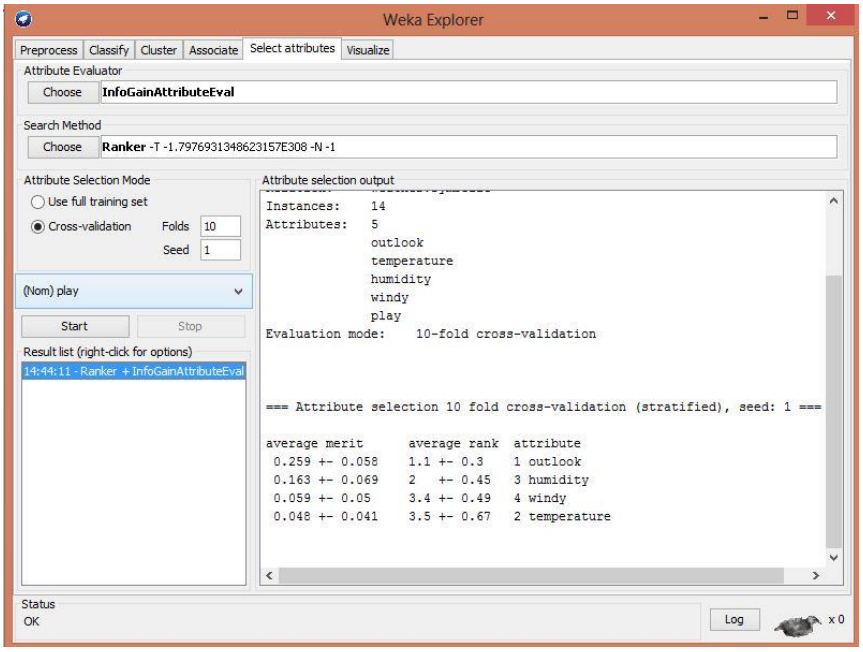
Bu ekranda veri kümesine birliktelik kuralları uygulanır. Veri kümesi içindeki genel eğilimin tespiti için kullanılan algoritmaları barındıran ekrandır.



Şekil 2.14 Weka Birliktelik Kuralları Ekranı

Seçim Özellikleri Ekranı (Select Attributes):

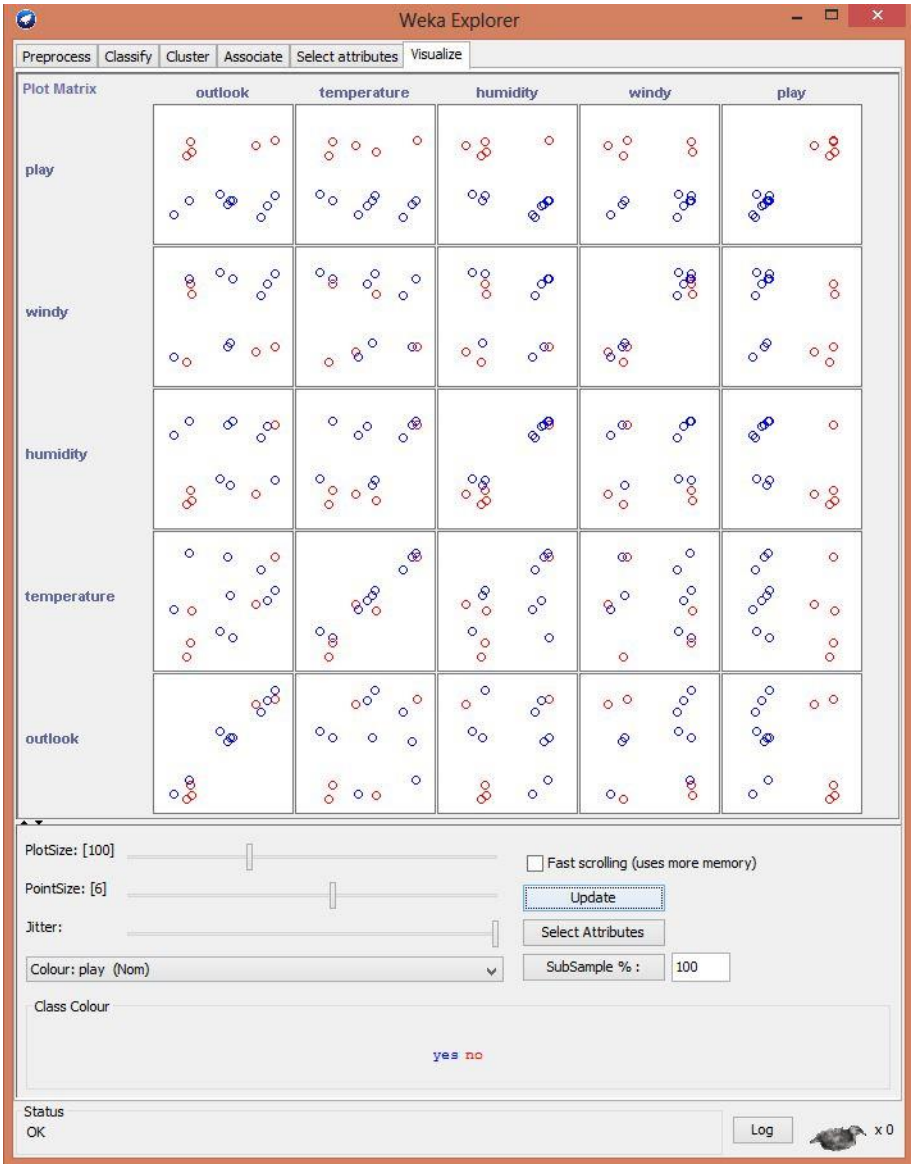
Veri kümesinde seçili olan özniteliğe etki eden diğer özniteliklerin etkisini nicel olarak gözlemlemek için kullanılır.



Şekil 2.15 Weka Özellik Seçimleri Ekranı

Görselleştirme Ekranı (Visualize):

Bu ekran ile farklı öznitelikler arasındaki ilişkileri grafik olarak görüntülemek mümkün olmaktadır. Genel görünüm yerine kullanıcı satır ve sütün isimlerinin birleştiği küçük demo pencerelerden istenilen bir tanesine tıklayarak özniteliği daha net görüp ele alabilir. Ekranda bulunan işaretçilerin boyutları, ekranın alt tarafındaki panel seçenekleri kullanılarak ayarlanabilir.



Şekil 2.16 Weka Görselleştirme Ekranı

BÖLÜM 3

MATERYAL ve YÖNTEM

3.1. Veri Toplama Araçlarının Hazırlanması, Verilerin Toplanması

Çalışmada kullanılan veri kümesinin eğitim alanlı bir veri kümesi oluşturulması amaçlanmıştır. Litertürde öğrencilerle yapılan çalışmalarda, veri kümesi kayıtlarının oluşturulmasında okullarda kullanılan öğrenci bilgi sistemlerinden, araştırmacının hazırladığı anketten veya testlerden yararlanıldığı görülmektedir.

Bu çalışmada veri kümesi oluşturulması için anket ve başarı testlerinden yapılması düşünülmüştür. Veri kümesinin oluşturulması aşamasında okullarda görevli rehber, öğretmen ve idareciler ile görüşülerek veri kümesinde kullanılabilecek öznitelikler hakkında görüşülmüş ve elde edilen bilgiler ışığında veri kümesi için kullanılacak öznitelik listesi oluşturulmuştur. Bu bağlamda Edirne ilinden bir saha çalışması gerçekleştirilmiştir. Bu saha çalışmasında Edirne merkezinde bulunan tüm ilköğretim okullarının 4.sınıfında okuyan öğrencilerin akademik başarıları ve kişisel bilgilerinin toplanması amaçlanmıştır.

Amaç doğrultusunda ilk olarak 2014-2015 eğitim yılı güz döneminde 4. sınıf Fen ve Teknoloji dersinin "Vücudumuz Bilmecesini Çözelim " ünitesi kavramları ile ilgili başarı testi uygulanması düşünülmüştür. Bu amaçla dersin öğretmenleri ile görüşülmüş, ders kitapları incelenmiş ve ilgili üniteyi kapsayan, kazanımları ölçen çoktan seçmeli sorular hazırlanmıştır. Başarıyı ölçmek için yeterli olduğu öğretmenlerce onaylanmış 20 soru seçilerek ünite başarı testi oluşturulmuştur. Aşağıda başarı testinde kullanılan sorular verilmiştir.

Vücudumuz Bilmecesini Çözelim Ünitesi Başarı Testi

1.Aşağıdakilerden hangisi iskelet sistemimizin bölümlerinden birisi değildir?

A) Kafatası  B) Omurga  C) Göğüs Kafesi  D) Kalp 

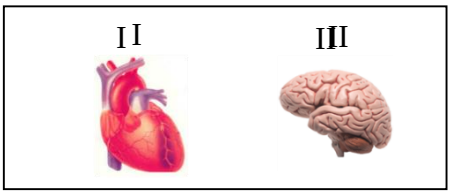
2.Aşağıdaki resimde I ve II ile gösterilen organları, koruyan iskelet kısımları hangi seçenekte doğru olarak yazılmıştır?

A) **I** Kafatası - **II** Omurga

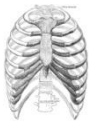
B) **I** Göğüs kafesi - **II** Kafatası

C) **I** Kafatası - **II** Göğüs kafesi

D) **I** Omurga - **II** Göğüs kafesi



3.Ali, bisikleti ile gezerken sert bir şekilde düşmüş ve yassı kemiklerinden birini kırmıştır. Kırılan yassı kemik aşağıdakilerden hangisinde olabilir?

A) El  B) Kol  C) Göğüs Kafesi  D) Bacak 

4.Aşağıdakilerden hangisi kemiklerin hareket etmesini sağlar?

A) Kas – Eklemler B) Deri - Kemik C) Yağ – Kas D) Kemik – Kalp



5.Aşağıdakilerden hangisi vücudumuzun kan dolaşım hızını arttırır?

A) Uyumak B) Oturmak C) Dinlenmek D) Egzersiz



6.Aşağıdakilerden hangisi dolaşım sistemini oluşturan yapı ve organlardan değildir?

A) Kalp B) Kan C) Damarlar D) Akciğer



7.Aşağıdakilerden hangisi sağlıklı bir insanın soluk alıp verme sıklığını artırmaz?

A) Koşmak B) Korkmak C) Heyecanlanmak D) Uyumak



8.Kalbin kasılıp gevşemesi ile damarlarda oluşturduğu etkiye ne denir?

A) Kalp Krizi B) Nabız C) Stres D) Kasılma



9.Aşağıdakilerden hangisi iskelet sistemimizde bulunan kemik şekillerinden biri değildir?



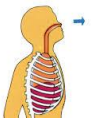
A) Uzun B) Oval C) Kısa D) Yassı

10.Kemiklerimizin hareketini sağlayan kas çiftleri ne şekilde kasılıp gevşer?



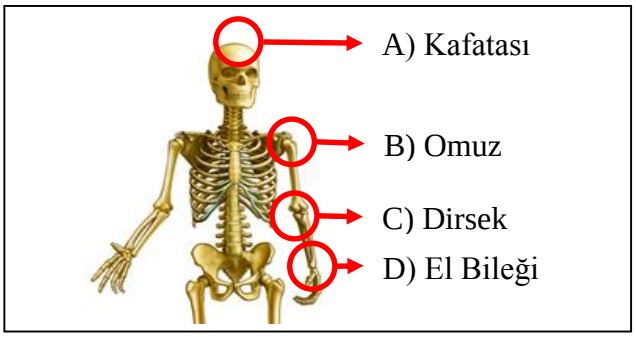
A) Aynı yönlü B) Zıt yönlü C) Çapraz D) Hiçbiri

11.Aşağıda, çevrede bulunan havanın solunum esnasında geçtiği organlar sıralanarak gösterilmiştir. Bu organ sıralamasında yanlış yerde olan organ hangi şıkta verilmiştir?



Hava ⇨ A) Burun ⇨ B) Yutak ⇨ C) Akciğer ⇨ D) Soluk Borusu

12.Aşağıda gösterilen alanlardan hangisi vücudumuzdaki hareketli eklemlerden değildir?



13.Aşağıdakilerden hangisi iskelet sisteminin görevlerinden birisi değildir?



A) Vücuda şekil verir. B) Vücudumuzun dik durmasını sağlar.

C) İç organları dış etkilerden korur. D) Kan pompalar.

14.İskelet sistemindeki kemiklerin, hareketli olmasını sağlayan bağlantı noktalarına ne denir?



A) Kas B) Yutak C) Eklem D) Omurga

15.Kanın vücutta dolaşmasını sağlayan yapılar aşağıdakilerden hangisinde doğru verilmiştir?



A) Kas-Kemik B) Kalp-Damar

C) Damar-Akciğer D) Kemik-Damar

16.Aşağıdakilerden hangisi, vücudumuzda kanı pompalayan organdır?



A) Kalp B) Akciğer C) Karaciğer D) Böbrek

17.Sabah uyandığında, "Boynum tutulmuş." diyen bir öğrencinin <u>sorunu</u> nedir?



A) Boyunda meydana gelen zedelenme B) Aşırı uyuma

C) Boyun kaslarının uzun süre kasılı kalması D) Yüz üstü yatmak

18.Spor salonunda koşan sağlıklı bir insanın organlarında nasıl bir değişim gözlenir?



A) Kalbi yavaşlar B) Solunum Yavaşlar

C) Kalbi hızlanır D) Kaslar sürekli gevşer

19.Aşağıdakilerden hangisi iskelet ve kas gelişimine <u>zarar verir</u>?



A) Okul çantasını sürekli aynı omuzda taşımak B)Yatakta uyumak

C) Bir eşyayı kaldırırken dizleri kırarak eğilmek D) Otururken dik durmak

20.Vücudumuzda kanı taşıma görevini hangi organımız yerine getirmektedir?



A) Damarlar B) Böbrekler C) Akciğerler D) Bağırsaklar

Ünite başarı testi öğrencilerin başlangıçtaki akademik bilgi düzeylerini tespit etmek için bu ünitenin işlenmesinden önce öntest olarak uygulanmıştır. Ünite bitiminde aynı başarı testi öğrencilere sontest olarak uygulanmıştır.

İkinci olarak öğrencilere ait kişisel bilgilerin toplanmasında bir anketten yararlanılmıştır. Anket hazırlanırken öğrencinin akademik başarısını etkileyebilecek çeşitli soruların ankette yer alması sağlanmıştır. Anket soruları hazırlanırken rehberlik servisinden destek alınmıştır. Uygulanan ankette yer alan soru sayısı 32'dir. Aşağıda öğrencilerin yanıtladığı anket soruları yer almaktadır.

Anket

1. Cinsiyetin nedir?

A) Kız

B) Erkek

C)

D)

2. Kaç yaşındasın?

A) 10

B) 11

C) 12

D) 13

3. Nerede yaşıyorsun?

A) Ailemin yanında

B) Yurtta

C) Akrabamın yanında

D) Hiçbiri

4. Kiminle yaşıyorsun?

A) Annem ile

B) Babam ile

C) Ailem ile

D) Diğer

5. Yaşadığın yerde kendine ait odan var mı?

A) Evet

B) Hayır

C)

D)

6. Annen yaşıyor mu?

A) Evet

B) Hayır

C)

D)

7. Annen çalışıyor mu?

A) Evet

B) Hayır

C)

D)

8. Annenin eğitim durumu nedir?

A) İlkokul

B) Ortaokul

C) Lise

D) Üniversite

9. Baban yaşıyor mu?

A) Evet

B) Hayır

C)

D)

10. Baban çalışıyor mu?

A) Evet

B) Hayır

C)

D)

11. Babanın eğitim durumu nedir?

A) İlkokul

B) Ortaokul

C) Lise

D) Üniversite

12. Kaç kardeşsiniz?

A) Tek Çocuğum

B) 2 kardeşiz

C) 3 kardeşiz

D) 4 ve daha fazla kardeşiz

13. Ailenizde özel durumu, herhangi bir rahatsızlık ya da engel durumu, olan biri var mı?

A) Annem

B) Babam

C) Kardeşim

D) Hiçbiri

14. Evde derslerine yardımcı olanlar aşağıdakilerden hangisidir?

A) Anne

B) Baba

C) Anne ve Baba

D) Diğer(Ağabey, abla vb.)

15. Annenin, babanın veya evinin telefon numarasını ezbere biliyor musun?

A) Evet

B) Hayır

C)

D)

16. Evinin adresini ezbere biliyor musun?

A) Evet

B) Hayır

C)

D)

17.Okula ulaşımını nasıl sağlıyorsun?

A) Okul Servisi ile

B) Araba ile

C) Yürüyerek

D) Minibüs ile

18. Okulda vakit geçirmeyi seviyor musun?

A) Hiçbir zaman

B) Bazen

C) Sık sık

D) Her zaman

19. En çok sevdiğin dersleri sıralasan ilk sırada aşağıdaki derslerden hangisi olur?

A) Fen ve Teknoloji

B) Matematik

C) Türkçe

D) Bunlardan Hiçbiri

20. Evde, derslerine çalışmak için vakit ayırıyor musun?

A) Hiçbir zaman

B) Bazen

C) Sık sık

D) Her zaman

21. Öğretmenlerinin verdiği ödevleri yapıyor musun?

A) Hiçbir zaman

B) Bazen

C) Sık sık

D) Her zaman

22. Teneffüslerde arkadaşların ile vakit geçirmekten hoşlanıyor musun?

A) Hiçbir zaman

B) Bazen

C) Sık sık

D) Her zaman

23. Yazın, okullar kapandığında tatile gidiyor musun?

A) Hiçbir zaman

B) Bazen

C) Sık sık

D) Her zaman

24. Kışın, okullar kapandığında tatile gidiyor musun?

A) Hiçbir zaman

B) Bazen

C) Sık sık

D) Her zaman

25. Kitap okumayı seviyor musun?

A) Hiçbir zaman

B) Bazen

C) Sık sık

D) Her zaman

26. Günde ortalama kaç saat bilgisayar kullanıyorsun?

A) Kullanmıyorum

B) 1-2 saat

C) 2-4 saat

D) 4 saatten fazla

27. Günde ortalama kaç saat televizyon izliyorsun?

A) İzlemiyorum

B) 1-2 saat

C) 2-4 saat

D) 4 saatten fazla

28. Evcil hayvanın var mı?

A) Var

B) Yok

C)

D)

29. Sıra arkadaşından memnun musun?

A) Memnun değilim

B) Bazen memnunum

C) Genelde memnunum

D) Çok memnunum

30. Okuldaki etkinliklerde görev alır mısın?

A) Hiçbir zaman

B) Bazen

C) Sık sık

D) Her zaman

31.Geceleri saat kaçta uyursun?

A) Saat 8 de

B) Saat 9 da

C) Saat 10 da

D) Saat 11 veya daha geç

32. Öğretmenlerinin isimlerini biliyor musun?

A) Hiçbirinin ismini bilmiyorum

B) Bir kaçının ismini biliyorum

C) Çoğunun ismini biliyorum

D) Hepsinin ismini biliyorum

Veri kümesinde yer alan her bir kayıt öğrencinin, test sonuçlarından ve anket sorularına verdiği yanıtlardan oluşturulmuştur.

Yapılan başarı testi ve anket uygulamasının sonuçları, öğrenciler tarafından optik forma işaretlenmiştir. Optik formalardaki veriler Microsoft Visual Studio 2010 C# kullanılarak geliştirilen program aracılığı ile Access veri tabanına kaydedilip saklanmıştır.

3.2. Veri Kümesi Önişlem

Verikaynağına uygulanan önişlem aşamasında, verinin gürültüden temizlenmesi, veri kümelerinin hazırlanması ve veri dönüştürme işlemlerinden bahsedilmiştir.

3.2.1. Veri Gürültüsünün Temizlenmesi

Veri gürültüsü, konu dışı ve anlamsız bilgi anlamına gelir. Gürültü, eğitim verisi açısından çok önemlidir. Eğer seçilen eğitim verisi gürültülü ise bu durumda yeni gelen verilerin tahmini zorlaşır [12].

Veritabanına kaydedilen veriler incelenerek anlamsız ve eksik bulunanların algoritma başarısındaki etkisi göz önüne alınarak veri kümesinden çıkarılmıştır; Bu temizleme işleminde Visual Studio 2010 C# kullanılarak geliştirilen program ve Weka programının Visualize ekranı olmak üzere iki araç kullanılmıştır.

1- Oluşturulan programda yazılan SQL cümlesi ile;
- Testi yanıtlayıp anketi cevaplamayan kayıtlar bulunmuş,
- Anketi yanıtlayıp testi cevaplamayan kayıtlar bulunmuş,
- Anne sağ olmamasına rağmen annmele yaşıyorum veya annemle ders çalışıyorum şıklarını seçen örnekler bulunmuş,
- Baba sağ olmamasına rağmen babamla yaşıyorum veya babamla ders çalışıyorum şıklarını seçen örnekler bulunmuş,
- Öntesti başarılı olup sontesti başarısız olan örnekler bulunup çıkarılmıştır.
- Veri kümesinde her bir örneğin sahip olduğu öznitelik sayısı 36 dır. Bu özniteliklerden herhangi birinde olacak boş değerin sonuca etkisinin olabileceği düşünülerek boş özniteliğe sahip örnekler veri kümesinden çıkarılmıştır.

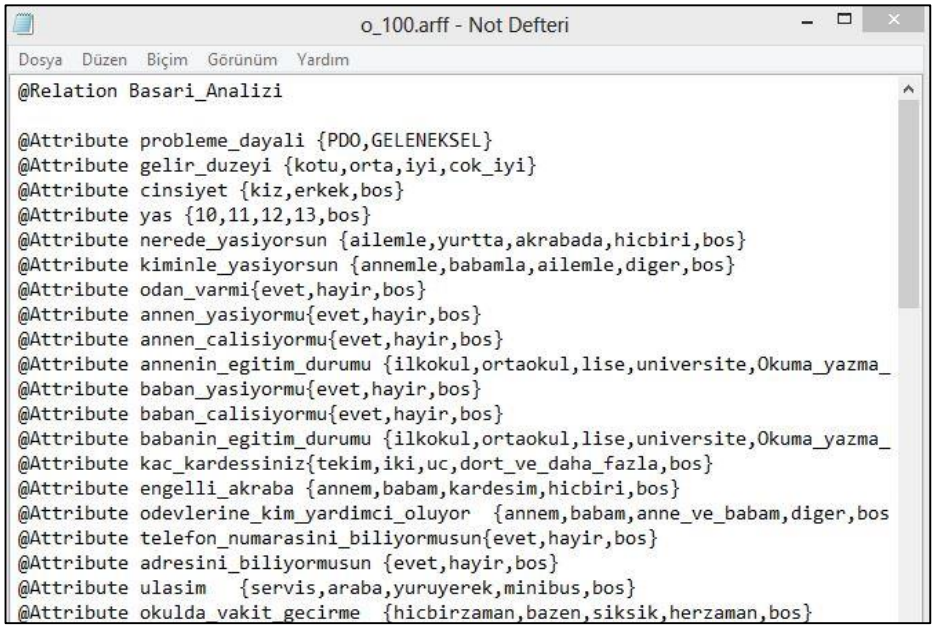
2- Weka Explorer penceresinden ulaşılan Visualize ekranı kullanılarak veri kümesindeki sontest özniteliğinin veri kümesindeki diğer tüm öznitelikler için

örnek dağılımı incelenmiştir. İnceleme sonrasında anlamsız olan örnekler bulunarak veri kümesinden çıkarılmıştır. Aşağıdaki resimde sontest ile yaş özniteliği örneklerinin dağılımı görüntülenmiştir. Bu ekranda ilgili işaretçi çift tıklanarak örneğin sıra numarası öğrenilmektedir. Öğrenilen sıra numarası geliştirilen program kullanılarak veri kümesinden silinmiştir.

3.2.2. Veri kümelerinin Hazırlanması

Temizlik sonrası 570 örneğe sahip bir kaynak veri kümesi elde edilmiştir. Bu bağlamda Microsoft Visual Studio 2010 c# programı kullanılarak geliştirilen program aracılığı ile kaynak veri kümesi içinden tekrarsız seçim yapılarak aşağıdaki alt veri kümeleri oluşturulmuştur.

- 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 550 ve 570 örneğe sahip yeni alt veri kümeleri oluşturulmuş ve arff dosya formatında kaydedilmiştir.
- Aşağıda geliştirilen programın oluşturduğu arff formatındaki bir veri kümesi dosyası gösterilmiştir.



Şekil 3.1 Arff Formatlı Veri kümesi Dosyası

3.2.3. Veri Dönüştürme

Çalışmada, classify value (sontest) özniteliğinin sahip olduğu sınıf sayısının algoritma başarısında etkili olabileceği düşüncesiyle sahip olabileceği farklı sayıdaki

sınıflar için dönüştürme işlemleri gerçekleştirilmiştir. Bu dönüştürme veri kümelerinin tümü için gerçekleştirilmiştir.

Sontest özniteliği ile ilgili dönüşümler;

1- Visual Studio 2010 programı kullanılarak yazılan programda oluşturulan veri kümelerinin tümünde, sontest özniteliğinin sahip olduğu (0,5,10,…100) 21 ayrı değer bulunmaktadır.

2- Sontest değeri sözel değerlendirme türüne dönüştürülerek 0-44 aralığı Başarısız, 45-54 aralığı Geçer, 55-59 aralığı Orta, 70-84 aralığı iyi, 85-100 aralığı pekiyi olarak dönüştürülmüş. Bu dönüştürme işlemi ilk durumda oluşturulmuş olan alt veri kümelerinin tümü üzerinde yapılmıştır.

Tablo 3.1 Not Dönüşüm

Not Aralığı	**Yeni Değer**
0-44	Başarısız
45-54	Geçer
55-59	Orta
70-84	İyi
85-100	Pekiyi

3- Sontest değeri için kullanılan diğer dönüşüm işlemi de çan eğrisi hesaplanırken eşik değerini bulmak için kullanılan ortalama alma işlemidir. Mevcut verikaynağı için hesaplanan sontest özniteliğinin ortalama değeri 80 olup bu değerin altında alınan puan için Başarısız, 80 ve üstü puan için Başarılı olarak dönüştürülmüş. Bu dönüştürme işlemi ilk durumda oluşturulmuş olan alt veri kümelerinin tümü üzerinde yapılmıştır. Ancak alınan sonuçlarda oluşturulan veri kümesinin dengesiz olduğu görülmüştür. Denemeler sonrasında başarı ve başarısız eşik değerinin 70 olmasına karar verilmiştir.

Tablo 3.2 Not Dönüşüm

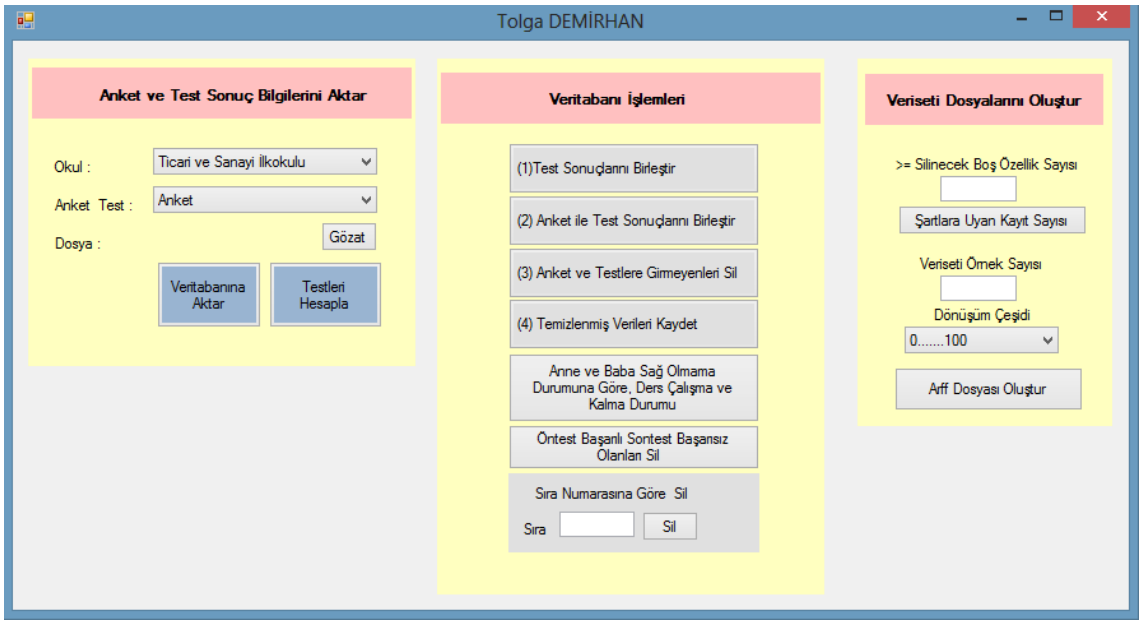
Not Aralığı	**Yeni Değer**
0-69	Başarısız
70-100	Başarılı

Çalışmada yapılan deneylerde dönüştürme işlemleri farklı öznitelikler üzerinde gerçekleştirilerek performans üzerindeki etkisi incelenmiştir.

3.3. Geliştirilen Program

Çalışmaya konu olan veri kümesinin oluşturulması noktasında en önemli adım bilginin katılımcılardan alınması sürecidir. Bu noktada gönüllü öğrencilerin farklı zamanlarda katıldıkları anket ve öntest-sontest başarı testi yanıtları anket ve testler yanında verilen optik formlar kullanılarak alınmıştır.

Daha sonra öğrencilerden alınan optik formların sahip oldukları bilgilerin dijital ortama aktarılması için ilk olarak bir optik okuyucudn faydalanılmıştır. Daha sonra bilgileri veritabanında kayıt altında tutmak ve ihtiyaçlar doğrultusunda veriler üzerinde ekleme, düzenleme, silme, filtreleme, .arff formatında aktarma vb. işlemleri yapabilmek için Microsoft Visual Studio 2010 C# programı kullanılarak bir uygulama yazılımı geliştirilmiştir. Aşağıda program arayüzünde kullanılan bölümler hakkında kısa açıklamalar ve kodlara yer verilecektir.



Şekil 3.2 Program Arayüzü

3.3.1. Anket ve Test Sonuçlarını Aktar



Şekil 3.3 Anket ve Test Sonuç Bilgilerini Aktar

Program arayüzünün **Anket ve Test Sonuç Bilgilerini Aktar** başlıklı bölümünde, **okul** başlıklı bilgi alanında Edirne ilinde çalışılan okulun adı, **anket-test** alanında okulun hangi çalışmasının (anket, öntest, sontest) veritabanına girileceği belirlendikten sonra **Gözat** komut düğmesine basılarak optik okuyucudan alınan bilgilerin oluşturulduğu dosyanın yeri seçilir.

Ayarlamalar bitirildikten sonra seçili olan okula ait test veya anket sonuçları **Veritabanına Aktar komut** düğmesine basılarak veritabanına aktarılmaktadır. Veri tabanına aktarılan veriler içerisinde öğrencilerin başarı testine verdiği yanıtlar da bulunmaktadır. Bu testin bir öğrenci için başarı notunun hesaplanması için bu bölümde veritabanına aktar komut düğmesinin yanında bulunan **Testleri Hesapla** komut düğmesine basılır. Testleri Hesapla komut düğmesine basıldıktan sonra aktarılan testlerin, cevap anahtarı kullanılarak hesaplanması yapılmaktadır.

Aşağıda **Veritabanına Aktar** komut düğmesi altında kullanılmış kodlar yer almaktadır.

```
private void btn_aktar_Click(object sender, EventArgs e)
{
if(cmb_anket_test.Text !="" && cmb_okul.Text !="" && lbl_dosya_yolu.Text !="")
{
OleDbConnection con = new OleDbConnection("Provider=Microsoft. ACE. OLEDB.12.0;Data
Source=data.accdb ");
con.Open();
List<string> list_abc = new List<string>();

StreamReader dosya = new StreamReader(lbl_dosya_yolu.Text );

while (!dosya.EndOfStream) // txt dosyasını döndürür
{
    list_abc.Add (dosya.ReadLine());
}
/// list içindekiler tabloya aktar
int uygun = 0;
string ogrenci_no = "";
```

```
for (int i=0;i<list_abc.Count ;i++)
{
    string satir_detay = list_abc[i].ToString();
    char[] karakterler = satir_detay.ToArray();
    // Öğrenci numarası sorunsuzmu
    for (int n = 0; n < 5; n++)
    {
        if (karakterler[n].ToString() == "*" || karakterler[n].ToString() == "")
        {
            uygun = uygun + 1;
        }
        else
        {
            ogrenci_no = ogrenci_no + karakterler[n].ToString();
        }
    }

    if (uygun == 0) // sorun yok kaydet
    {
            // ANKET KAYDET
        if (cmb_anket_test.SelectedIndex == 0)
        {
string anket_ekle = "insert into anket (okul_id,ogrenci_no, s1,s2,s3,s4,s5,s6,s7,s8, s9,s10,s11,
s12,s13,s14,s15,s16,s17,s18,s19,s20,s21,s22,s23,s24,s25,s26,s27,s28,s29,s30,s31,s32)      values
(@okul_id,@ogrenci_no,@s1,@s2,@s3,@s4,@s5,@s6,@s7,@s8,@s9,@s10,@s11,@s12,@s13,@s14,@s15,@s16,@s1
7,@s18,@s19,@s20,@s21,@s22,@s23,@s24,@s25,@s26,@s27,@s28,@s29,@s30,@s31,@s32)";
            OleDbCommand cmd = new OleDbCommand(anket_ekle,con);
            cmd.Parameters.AddWithValue("@okul_id",cmb_okul.SelectedValue );
            cmd.Parameters.AddWithValue("@ogreni_no", ogrenci_no );
            cmd.Parameters.AddWithValue("@s1", karakterler[6].ToString ());
            cmd.Parameters.AddWithValue("@s2", karakterler[7].ToString());
            cmd.Parameters.AddWithValue("@s3", karakterler[8].ToString());
            cmd.Parameters.AddWithValue("@s4", karakterler[9].ToString());
            cmd.Parameters.AddWithValue("@s5", karakterler[10].ToString());
            cmd.Parameters.AddWithValue("@s6", karakterler[11].ToString());
            cmd.Parameters.AddWithValue("@s7", karakterler[12].ToString());
            cmd.Parameters.AddWithValue("@s8", karakterler[13].ToString());
            cmd.Parameters.AddWithValue("@s9", karakterler[14].ToString());
            cmd.Parameters.AddWithValue("@s10", karakterler[15].ToString());
            cmd.Parameters.AddWithValue("@s11", karakterler[16].ToString());
            cmd.Parameters.AddWithValue("@s12", karakterler[17].ToString());
            cmd.Parameters.AddWithValue("@s13", karakterler[18].ToString());
            cmd.Parameters.AddWithValue("@s14", karakterler[19].ToString());
            cmd.Parameters.AddWithValue("@s15", karakterler[20].ToString());
            cmd.Parameters.AddWithValue("@s16", karakterler[21].ToString());
            cmd.Parameters.AddWithValue("@s17", karakterler[22].ToString());
            cmd.Parameters.AddWithValue("@s18", karakterler[23].ToString());
            cmd.Parameters.AddWithValue("@s19", karakterler[24].ToString());
            cmd.Parameters.AddWithValue("@s20", karakterler[25].ToString());
            cmd.Parameters.AddWithValue("@s21", karakterler[26].ToString());
            cmd.Parameters.AddWithValue("@s22", karakterler[27].ToString());
            cmd.Parameters.AddWithValue("@s23", karakterler[28].ToString());
            cmd.Parameters.AddWithValue("@s24", karakterler[29].ToString());
            cmd.Parameters.AddWithValue("@s25", karakterler[30].ToString());
            cmd.Parameters.AddWithValue("@s26", karakterler[31].ToString());
            cmd.Parameters.AddWithValue("@s27", karakterler[32].ToString());
            cmd.Parameters.AddWithValue("@s28", karakterler[33].ToString());
            cmd.Parameters.AddWithValue("@s29", karakterler[34].ToString());
            cmd.Parameters.AddWithValue("@s30", karakterler[35].ToString());
            cmd.Parameters.AddWithValue("@s31", karakterler[36].ToString());
            cmd.Parameters.AddWithValue("@s32", karakterler[37].ToString());
            cmd.ExecuteNonQuery();

        }
// ÖNTEST KAYDET

if (cmb_anket_test.SelectedIndex == 1)
{
string anket_ekle = "insert into ontest (okul_id,ogrenci_no,s1,s2,s3,s4,s5,s6,s7,s8, s9,s10,
s11,s12,s13,s14,s15,s16,s17,s18,s19,s20) values(@okul_id,@ogrenci_no, @s1,@s2,@s3,@s4,
@s5,@s6,@s7, @s8,@s9, @s10,@s11, @s12,@s13, @s14,@s15,@s16,@s17,@s18,@s19,@s20)";
```

```
            OleDbCommand cmd = new OleDbCommand(anket_ekle, con);
            cmd.Parameters.AddWithValue("@okul_id", cmb_okul.SelectedValue);
            cmd.Parameters.AddWithValue("@ogreni_no", ogrenci_no);
            cmd.Parameters.AddWithValue("@s1", karakterler[6].ToString());
            cmd.Parameters.AddWithValue("@s2", karakterler[7].ToString());
            cmd.Parameters.AddWithValue("@s3", karakterler[8].ToString());
            cmd.Parameters.AddWithValue("@s4", karakterler[9].ToString());
            cmd.Parameters.AddWithValue("@s5", karakterler[10].ToString());
            cmd.Parameters.AddWithValue("@s6", karakterler[11].ToString());
            cmd.Parameters.AddWithValue("@s7", karakterler[12].ToString());
            cmd.Parameters.AddWithValue("@s8", karakterler[13].ToString());
            cmd.Parameters.AddWithValue("@s9", karakterler[14].ToString());
            cmd.Parameters.AddWithValue("@s10", karakterler[15].ToString());
            cmd.Parameters.AddWithValue("@s11", karakterler[16].ToString());
            cmd.Parameters.AddWithValue("@s12", karakterler[17].ToString());
            cmd.Parameters.AddWithValue("@s13", karakterler[18].ToString());
            cmd.Parameters.AddWithValue("@s14", karakterler[19].ToString());
            cmd.Parameters.AddWithValue("@s15", karakterler[20].ToString());
            cmd.Parameters.AddWithValue("@s16", karakterler[21].ToString());
            cmd.Parameters.AddWithValue("@s17", karakterler[22].ToString());
            cmd.Parameters.AddWithValue("@s18", karakterler[23].ToString());
            cmd.Parameters.AddWithValue("@s19", karakterler[24].ToString());
            cmd.Parameters.AddWithValue("@s20", karakterler[25].ToString());
            cmd.ExecuteNonQuery();
        }
        // SONTEST KAYDET
if (cmb_anket_test.SelectedIndex == 2)
{
string anket_ekle = "insert into sontest
(okul_id,ogrenci_no,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19,s20)
values(@okul_id,@ogrenci_no,@s1,@s2,@s3,@s4,@s5,@s6,@s7,@s8,@s9,@s10,@s11,@s12,@s13,@s14,@s15,@s
16,@s17,@s18,@s19,@s20)";
            OleDbCommand cmd = new OleDbCommand(anket_ekle, con);
            cmd.Parameters.AddWithValue("@okul_id", cmb_okul.SelectedValue);
            cmd.Parameters.AddWithValue("@ogreni_no", ogrenci_no);
            cmd.Parameters.AddWithValue("@s1", karakterler[6].ToString());
            cmd.Parameters.AddWithValue("@s2", karakterler[7].ToString());
            cmd.Parameters.AddWithValue("@s3", karakterler[8].ToString());
            cmd.Parameters.AddWithValue("@s4", karakterler[9].ToString());
            cmd.Parameters.AddWithValue("@s5", karakterler[10].ToString());
            cmd.Parameters.AddWithValue("@s6", karakterler[11].ToString());
            cmd.Parameters.AddWithValue("@s7", karakterler[12].ToString());
            cmd.Parameters.AddWithValue("@s8", karakterler[13].ToString());
            cmd.Parameters.AddWithValue("@s9", karakterler[14].ToString());
            cmd.Parameters.AddWithValue("@s10", karakterler[15].ToString());
            cmd.Parameters.AddWithValue("@s11", karakterler[16].ToString());
            cmd.Parameters.AddWithValue("@s12", karakterler[17].ToString());
            cmd.Parameters.AddWithValue("@s13", karakterler[18].ToString());
            cmd.Parameters.AddWithValue("@s14", karakterler[19].ToString());
            cmd.Parameters.AddWithValue("@s15", karakterler[20].ToString());
            cmd.Parameters.AddWithValue("@s16", karakterler[21].ToString());
            cmd.Parameters.AddWithValue("@s17", karakterler[22].ToString());
            cmd.Parameters.AddWithValue("@s18", karakterler[23].ToString());
            cmd.Parameters.AddWithValue("@s19", karakterler[24].ToString());
            cmd.Parameters.AddWithValue("@s20", karakterler[25].ToString());
            cmd.ExecuteNonQuery();
        }
    }
    uygun = 0;
    ogrenci_no = "";
}
con.Close ();
MessageBox.Show("Aktarıldı");
}
else
{
MessageBox.Show ("Alanları boş geçemezsin");
}
}
```

Aşağıda **Testleri Hesapla** komut düğmesi altında kullanılmış kodlar yer almaktadır.

```
private void btn_hesapla_Click(object sender, EventArgs e)
{
OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=data.accdb ");
con.Open();

string ontest="select * from ontest";
OleDbDataAdapter dap=new OleDbDataAdapter (ontest,con );
DataTable dt=new DataTable ();
dap.Fill (dt);
int notumuz = 0;
for (int i = 0; i < dt.Rows.Count; i++)
{
if (dt.Rows[i]["s1"].ToString() == "D") notumuz = notumuz + 5;
if (dt.Rows[i]["s2"].ToString() == "B") notumuz = notumuz + 5;
if (dt.Rows[i]["s3"].ToString() == "C") notumuz = notumuz + 5;
if (dt.Rows[i]["s4"].ToString() == "A") notumuz = notumuz + 5;
if (dt.Rows[i]["s5"].ToString() == "D") notumuz = notumuz + 5;
if (dt.Rows[i]["s6"].ToString() == "B") notumuz = notumuz + 5;
if (dt.Rows[i]["s7"].ToString() == "D") notumuz = notumuz + 5;
if (dt.Rows[i]["s8"].ToString() == "B") notumuz = notumuz + 5;
if (dt.Rows[i]["s9"].ToString() == "B") notumuz = notumuz + 5;
if (dt.Rows[i]["s10"].ToString() == "B") notumuz = notumuz + 5;
if (dt.Rows[i]["s11"].ToString() == "C") notumuz = notumuz + 5;
if (dt.Rows[i]["s12"].ToString() == "A") notumuz = notumuz + 5;
if (dt.Rows[i]["s13"].ToString() == "D") notumuz = notumuz + 5;
if (dt.Rows[i]["s14"].ToString() == "C") notumuz = notumuz + 5;
if (dt.Rows[i]["s15"].ToString() == "B") notumuz = notumuz + 5;
if (dt.Rows[i]["s16"].ToString() == "A") notumuz = notumuz + 5;
if (dt.Rows[i]["s17"].ToString() == "C") notumuz = notumuz + 5;
if (dt.Rows[i]["s18"].ToString() == "C") notumuz = notumuz + 5;
if (dt.Rows[i]["s19"].ToString() == "A") notumuz = notumuz + 5;
if (dt.Rows[i]["s20"].ToString() == "A") notumuz = notumuz + 5;

string guncelle = "update ontest set puan=@a where id=@b";
OleDbCommand cmd = new OleDbCommand(guncelle,con );
cmd.Parameters.AddWithValue("@a",notumuz );
cmd.Parameters.AddWithValue("@b",dt.Rows [i]["id"].ToString ());
cmd.ExecuteNonQuery();
notumuz = 0;
}
//// sontest

string sontest = "select * from sontest";
OleDbDataAdapter sdap = new OleDbDataAdapter(sontest, con);
DataTable sdt = new DataTable();
sdap.Fill(sdt);
int snotumuz = 0;
for (int i = 0; i < sdt.Rows.Count; i++)
{
if (sdt.Rows[i]["s1"].ToString() == "D") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s2"].ToString() == "B") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s3"].ToString() == "C") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s4"].ToString() == "A") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s5"].ToString() == "D") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s6"].ToString() == "B") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s7"].ToString() == "D") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s8"].ToString() == "B") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s9"].ToString() == "B") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s10"].ToString() == "B") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s11"].ToString() == "C") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s12"].ToString() == "A") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s13"].ToString() == "D") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s14"].ToString() == "C") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s15"].ToString() == "B") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s16"].ToString() == "A") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s17"].ToString() == "C") snotumuz = snotumuz + 5;
```

```
if (sdt.Rows[i]["s18"].ToString() == "C") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s19"].ToString() == "A") snotumuz = snotumuz + 5;
if (sdt.Rows[i]["s20"].ToString() == "A") snotumuz = snotumuz + 5;

string sguncelle = "update sontest set puan=@a where id=@b";
OleDbCommand scmd = new OleDbCommand(sguncelle, con);
scmd.Parameters.AddWithValue("@a", snotumuz);
scmd.Parameters.AddWithValue("@b", sdt.Rows[i]["id"].ToString());
scmd.ExecuteNonQuery();
snotumuz = 0;
}
con.Close();
}
```

3.3.2. Veritabanı İşlemleri



Şekil 3.4 Veritabanı İşlemleri

Bu bölüm, verilerin öğrenci merkezli birleştirilmesi, örneklerin düzenlenmesi, koşulları sağlamayanların veritabanından çıkarılması için kullanılan komutları barındırır. Bölümde yer alan komut düğmelerinin program içindeki görevleri aşağıda kısaca açıklanmıştır.

(1) Test Sonuçlarını Birleştir komut düğmesi kullanılarak, veritabanına aktarılan bilgilerden öntest ve sontest başarı sonuçlarının öğrenci ile ilişkilendirilmesi sağlanır. **(2) Anket ve Test Sonuçlarını Birleştir** komut düğmesi kullanılarak öğrencinin bir önceki komut düğmesi kullanılarak ilişkilendirdiği test sonuçlarına ankete verdiği yanıtlarda ilişkilendirilir. Anket ve başarı testini içeren saha çalışmada üzerinde titizlikle durulan

konulardan biri oluşturulacak veri kümesindeki öğrencilerin yapılan öntest, sontest ve anketlere katılmış olmalarıdır. Bu koşulu sağlamayan öğrencileri veritabanından çıkarmak için **(3) Ankete ve Testlere Girmeyenleri Sil** başlıklı komut düğmesi kullanılmaktadır. Yukarıdaki (1), (2) ve (3) adımları gerçekleştirildikten sonra elde edilen ilişkiler kullanılarak oluşturulan yeni tabloya ilişkili kayıtlar aktarılır. Bu işlem için **(4) Temizlenmiş Verileri Kaydet** komut düğmesi kullanılır.

Aşağıda **(1) Test Sonuçlarını Birleştir** komut düğmesi altında kullanılan kaynak kodlar verilmiştir.

```
private void button1_Click(object sender, EventArgs e)
{
OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=data.accdb ");
con.Open();

string listele="select * from ontest";
OleDbDataAdapter dap=new OleDbDataAdapter (listele,con );
DataTable dt=new DataTable ();
dap.Fill (dt);

for (int i = 0; i < dt.Rows.Count; i++)
{
string bul_ontest_puan = "select * from sontest where okul_id=@a and ogrenci_no=@b";
OleDbCommand cmd = new OleDbCommand(bul_ontest_puan,con );
cmd.Parameters.AddWithValue("@a",dt.Rows [i]["okul_id"].ToString ());
cmd.Parameters.AddWithValue("@b", dt.Rows[i]["ogrenci_no"].ToString());
OleDbDataAdapter dapx=new OleDbDataAdapter (cmd);
DataTable dtx=new DataTable ();
dapx.Fill (dtx );
if (dtx.Rows.Count > 0)
{
    string son_test_puan_ekle = "update ontest set son_test_puan=@a where id=@b";
    OleDbCommand cmd_update = new OleDbCommand(son_test_puan_ekle,con );
    cmd_update .Parameters.AddWithValue("@a", dtx.Rows[0]["puan"].ToString());
    cmd_update .Parameters.AddWithValue("@b", dt.Rows[i]["id"].ToString());
    cmd_update.ExecuteNonQuery();
}

}
string listeley = "select okul_id,ogrenci_no,puan,son_test_puan  from ontest";
OleDbDataAdapter dapy = new OleDbDataAdapter(listeley,con );
DataTable dty = new DataTable();
dapy.Fill(dty);

this.Text = dty.Rows.Count.ToString();
con.Close();
}
```

Aşağıda **(2) Anket ve Test Sonuçlarını Birleştir** komut düğmesi altında kullanılan kaynak kodlar verilmiştir.

```
private void button3_Click(object sender, EventArgs e)
{
OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=data.accdb ");
con.Open();

string anketler = "select * from anket";
OleDbDataAdapter dap = new OleDbDataAdapter(anketler,con );
```

```
DataTable dt = new DataTable();
dap.Fill(dt );
int varmi=0;

for (int i = 0; i < dt.Rows.Count; i++)
{
string bul_ontest = "select *  from ontest where okul_id=@a  and ogrenci_no=@b ";
OleDbCommand cmd = new OleDbCommand(bul_ontest,con );
cmd.Parameters.AddWithValue("@a",dt.Rows [i]["okul_id"].ToString ());
cmd.Parameters.AddWithValue("@b", dt.Rows[i]["ogrenci_no"].ToString());

OleDbDataAdapter dap_ontest = new OleDbDataAdapter(cmd );
DataTable dt_ontest = new DataTable();
dap_ontest.Fill(dt_ontest );

string okul_bul = "select * from okul where id=@a";
OleDbCommand cmd_okul = new OleDbCommand(okul_bul,con );
cmd_okul.Parameters.AddWithValue("@a",dt.Rows[i]["okul_id"].ToString ());
OleDbDataAdapter dap_okul = new OleDbDataAdapter(cmd_okul );
DataTable dt_okul = new DataTable();
dap_okul.Fill(dt_okul );

if (dt_ontest.Rows.Count >0)
{
    string anket_guncelle = "update anket set probleme_dayali=@x,ontest=@a, sontest=@b where
id=@c";
    OleDbCommand cmd_anket = new OleDbCommand(anket_guncelle,con );
    cmd_anket.Parameters.AddWithValue("@x", dt_okul .Rows[0]["probleme_dayali"].ToString());
    cmd_anket.Parameters.AddWithValue("@a",dt_ontest.Rows [0]["puan"]);
    cmd_anket.Parameters.AddWithValue("@b", dt_ontest.Rows[0]["son_test_puan"]);
    cmd_anket.Parameters.AddWithValue("@c", dt.Rows[i]["id"].ToString());
    cmd_anket.ExecuteNonQuery();
}

}

this.Text = dt.Rows.Count.ToString();
con.Close();
}
```

Aşağıda **(3) Ankete ve Testlere Girmeyenleri Sil** komut düğmesi altında kullanılan kaynak kodlar verilmiştir.

```
private void button4_Click(object sender, EventArgs e)
{
OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=data.accdb ");
con.Open();

string anketler = "delete * from anket where (sontest is null) or probleme_dayali is  null ";
OleDbCommand cmd = new OleDbCommand(anketler, con);
cmd.ExecuteNonQuery();

con.Close();
}
```

Aşağıda **(4) Temizlenmiş Verileri Kaydet** komut düğmesi altında kullanılan kaynak kodlar verilmiştir.

```
private void btn_anket_aktar_Click(object sender, EventArgs e)
{
OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=data.accdb ");
con.Open();
```

```
string anket_liste = "select * from anket where   (sontest is not null) and (probleme_dayali is
not null) ";
OleDbCommand cmdx = new OleDbCommand(anket_liste, con);
OleDbDataAdapter dap = new OleDbDataAdapter(cmdx );
DataTable dt=new DataTable ();
dap.Fill (dt);

string anket_aktar_sil = "delete * from anket_aktar";
OleDbCommand cmd_sil = new OleDbCommand(anket_aktar_sil,con );
cmd_sil.ExecuteNonQuery();

string cinsiyet = "";
string yas = "";
string nerede_yasiyorsun = "";
string kiminle_yasiyorsun = "";
string odan_varmi = "";
string annen_sagmi = "";
string annen_calisiyormu = "";
string annenin_egitim_durumu = "";
string baban_sagmi = "";
string baban_calisiyormu = "";
string babanin_egitim_durumu = "";
string kac_kardessiniz = "";
string engelli_akraba = "";
string odevlerine_kim_yardimci_oluyor = "";
string telefon_numarasini_biliyormusun = "";
string adresini_biliyormusun = "";
string ulasim = "";
string okulda_vakit_gecirme = "";
string en_coksevdigin_ders = "";
string ders_calisma_vakti = "";
string odev_yapıyormusun = "";
string arakadaslar_ile_vakit_gecirme = "";
string yaz_tatili = "";
string kis_tatili = "";
string kitap_okumayi_seviyormusun = "";
string kac_saat_bilgisayar_kullanıyorsun = "";
string kac_saat_tv = "";
string evcil_hayvan = "";
string sira_arakadasindan_memnunumusun = "";
string okul_etkinliginde_gorev = "";
string uyku_saati = "";
string ogretmen_isimlerini_biliyormusun = "";

for (int i = 0; i < dt.Rows.Count; i++)
{
string okul = "select * from okul where id=@a";
OleDbCommand cmd_okul = new OleDbCommand(okul, con);
cmd_okul.Parameters.AddWithValue("@a", dt.Rows[i]["okul_id"].ToString ());
OleDbDataAdapter dap_okul = new OleDbDataAdapter(cmd_okul );
DataTable dt_okul = new DataTable();
dap_okul.Fill(dt_okul );

// 1
if (dt.Rows[i]["s1"].ToString() == "A")
    cinsiyet = "kiz";
else if (dt.Rows[i]["s1"].ToString() == "B")
    cinsiyet = "erkek";
else cinsiyet = "bos";

// 2
if (dt.Rows[i]["s2"].ToString() == "A")
    yas  = "10";
else if (dt.Rows[i]["s2"].ToString() == "B")
    yas  = "11";
else yas = "bos";

// 3
if (dt.Rows[i]["s3"].ToString() == "A")
    nerede_yasiyorsun = "ailemle";
else if (dt.Rows[i]["s3"].ToString() == "B")
```

```
    nerede_yasiyorsun = "yurtta";
else if (dt.Rows[i]["s3"].ToString() == "C")
    nerede_yasiyorsun = "akrabada";
else if (dt.Rows[i]["s3"].ToString() == "D")
    nerede_yasiyorsun = "hicbiri";
else nerede_yasiyorsun = "bos";

// 4
if (dt.Rows[i]["s4"].ToString() == "A")
    kiminle_yasiyorsun = "annemle";
else if (dt.Rows[i]["s4"].ToString() == "B")
    kiminle_yasiyorsun = "babamla";
else if (dt.Rows[i]["s4"].ToString() == "C")
    kiminle_yasiyorsun = "ailemle";
else if (dt.Rows[i]["s4"].ToString() == "D")
    kiminle_yasiyorsun = "diger";
else kiminle_yasiyorsun = "bos";

////5
if (dt.Rows[i]["s5"].ToString() == "A")
    odan_varmi = "evet";
else if (dt.Rows[i]["s5"].ToString() == "B")
    odan_varmi  = "hayir";
else odan_varmi = "bos";

//6
if (dt.Rows[i]["s6"].ToString() == "A")
    annen_sagmi  = "evet";
else if (dt.Rows[i]["s6"].ToString() == "B")
    annen_sagmi  = "hayir";
else annen_sagmi = "bos";

////  7
if (dt.Rows[i]["s7"].ToString() == "A")
    annen_calisiyormu  = "evet";
else if (dt.Rows[i]["s7"].ToString() == "B")
    annen_calisiyormu= "hayir";
else annen_calisiyormu = "bos";

/// 8
if (dt.Rows[i]["s8"].ToString() == "A")
    annenin_egitim_durumu = "ilkokul";
else if (dt.Rows[i]["s8"].ToString() == "B")
    annenin_egitim_durumu = "ortaokul";
else if (dt.Rows[i]["s8"].ToString() == "C")
    annenin_egitim_durumu = "lise";
else if (dt.Rows[i]["s8"].ToString() == "D")
    annenin_egitim_durumu = "universite";
else annenin_egitim_durumu = "bos";

//9
if (dt.Rows[i]["s9"].ToString() == "A")
    baban_sagmi  = "evet";
else if (dt.Rows[i]["s9"].ToString() == "B")
    baban_sagmi  = "hayir";
else baban_sagmi = "bos";

////  10
if (dt.Rows[i]["s10"].ToString() == "A")
    baban_calisiyormu = "evet";
else if (dt.Rows[i]["s10"].ToString() == "B")
    baban_calisiyormu = "hayir";

else baban_calisiyormu = "bos";

/// 11
if (dt.Rows[i]["s11"].ToString() == "A")
    babanin_egitim_durumu  = "ilkokul";
else if (dt.Rows[i]["s11"].ToString() == "B")
    babanin_egitim_durumu = "ortaokul";
else if (dt.Rows[i]["s11"].ToString() == "C")
```

```
    babanin_egitim_durumu = "lise";
else if (dt.Rows[i]["s11"].ToString() == "D")
    babanin_egitim_durumu = "universite";
else babanin_egitim_durumu = "bos";

// 12
if (dt.Rows[i]["s12"].ToString() == "A")
    kac_kardessiniz = "tekim";
else if (dt.Rows[i]["s12"].ToString() == "B")
    kac_kardessiniz = "iki";
else if (dt.Rows[i]["s12"].ToString() == "C")
    kac_kardessiniz = "uc";
else if (dt.Rows[i]["s12"].ToString() == "D")
    kac_kardessiniz = "dort_ve_daha_fazla";
else kac_kardessiniz = "bos";

//13
if (dt.Rows[i]["s13"].ToString() == "A")
    engelli_akraba = "annem";
else if (dt.Rows[i]["s13"].ToString() == "B")
    engelli_akraba = "babam";
else if (dt.Rows[i]["s13"].ToString() == "C")
    engelli_akraba = "kardesim";
else if (dt.Rows[i]["s13"].ToString() == "D")
    engelli_akraba = "hicbiri";
else engelli_akraba = "bos";

// 14
if (dt.Rows[i]["s14"].ToString() == "A")
    odevlerine_kim_yardimci_oluyor= "annem";
else if (dt.Rows[i]["s14"].ToString() == "B")
    odevlerine_kim_yardimci_oluyor = "babam";
else if (dt.Rows[i]["s14"].ToString() == "C")
    odevlerine_kim_yardimci_oluyor = "anne_ve_babam";
else if (dt.Rows[i]["s14"].ToString() == "D")
    odevlerine_kim_yardimci_oluyor= "diger";
else odevlerine_kim_yardimci_oluyor = "bos";

// 15
if (dt.Rows[i]["s15"].ToString() == "A")
    telefon_numarasini_biliyormusun= "evet";
else if (dt.Rows[i]["s15"].ToString() == "B")
    telefon_numarasini_biliyormusun= "hayir";
else telefon_numarasini_biliyormusun = "bos";

// 16
if (dt.Rows[i]["s16"].ToString() == "A")
    adresini_biliyormusun = "evet";
else if (dt.Rows[i]["s16"].ToString() == "B")
    adresini_biliyormusun= "hayir";
else adresini_biliyormusun = "bos";

//17
if (dt.Rows[i]["s17"].ToString() == "A")
    ulasim = "servis";
else if (dt.Rows[i]["s17"].ToString() == "B")
    ulasim = "araba";
else if (dt.Rows[i]["s17"].ToString() == "C")
    ulasim = "yuruyerek";
else if (dt.Rows[i]["s17"].ToString() == "D")
    ulasim = "minibus";
else ulasim = "bos";

//18
if (dt.Rows[i]["s18"].ToString() == "A")
    okulda_vakit_gecirme = "hicbirzaman";
else if (dt.Rows[i]["s18"].ToString() == "B")
    okulda_vakit_gecirme = "bazen";
else if (dt.Rows[i]["s18"].ToString() == "C")
    okulda_vakit_gecirme = "siksik";
else if (dt.Rows[i]["s18"].ToString() == "D")
```

```
    okulda_vakit_gecirme = "herzaman";
else okulda_vakit_gecirme = "bos";

//19
if (dt.Rows[i]["s19"].ToString() == "A")
    en_coksevdigin_ders = "fenveteknoloji";
else if (dt.Rows[i]["s19"].ToString() == "B")
    en_coksevdigin_ders = "matematik";
else if (dt.Rows[i]["s19"].ToString() == "C")
    en_coksevdigin_ders = "turkce";
else if (dt.Rows[i]["s19"].ToString() == "D")
    en_coksevdigin_ders = "hicbiri";
else en_coksevdigin_ders = "bos";

//20
if (dt.Rows[i]["s20"].ToString() == "A")
    ders_calisma_vakti = "hicbirzaman";
else if (dt.Rows[i]["s20"].ToString() == "B")
    ders_calisma_vakti = "bazen";
else if (dt.Rows[i]["s20"].ToString() == "C")
    ders_calisma_vakti = "siksik";
else if (dt.Rows[i]["s20"].ToString() == "D")
    ders_calisma_vakti = "herzaman";
else ders_calisma_vakti = "bos";

// 21
if (dt.Rows[i]["s21"].ToString() == "A")
    odev_yapıyormusun = "hicbirzaman";
else if (dt.Rows[i]["s21"].ToString() == "B")
    odev_yapıyormusun = "bazen";
else if (dt.Rows[i]["s21"].ToString() == "C")
    odev_yapıyormusun = "siksik";
else if (dt.Rows[i]["s21"].ToString() == "D")
    odev_yapıyormusun = "herzaman";
else odev_yapıyormusun = "bos";

// 22
if (dt.Rows[i]["s22"].ToString() == "A")
    arakadaslar_ile_vakit_gecirme = "hicbirzaman";
else if (dt.Rows[i]["s22"].ToString() == "B")
    arakadaslar_ile_vakit_gecirme = "bazen";
else if (dt.Rows[i]["s22"].ToString() == "C")
    arakadaslar_ile_vakit_gecirme = "siksik";
else if (dt.Rows[i]["s22"].ToString() == "D")
    arakadaslar_ile_vakit_gecirme = "herzaman";
else arakadaslar_ile_vakit_gecirme = "bos";

//23
if (dt.Rows[i]["s23"].ToString() == "A")
    yaz_tatili = "hicbirzaman";
else if (dt.Rows[i]["s23"].ToString() == "B")
    yaz_tatili = "bazen";
else if (dt.Rows[i]["s23"].ToString() == "C")
    yaz_tatili = "siksik";
else if (dt.Rows[i]["s23"].ToString() == "D")
    yaz_tatili = "herzaman";
else yaz_tatili = "bos";

//24
if (dt.Rows[i]["s24"].ToString() == "A")
    kis_tatili = "hicbirzaman";
else if (dt.Rows[i]["s24"].ToString() == "B")
    kis_tatili = "bazen";
else if (dt.Rows[i]["s24"].ToString() == "C")
    kis_tatili = "siksik";
else if (dt.Rows[i]["s24"].ToString() == "D")
    kis_tatili = "herzaman";
else kis_tatili = "bos";

//25
if (dt.Rows[i]["s25"].ToString() == "A")
```

```
    kitap_okumayi_seviyormusun = "hicbirzaman";
else if (dt.Rows[i]["s25"].ToString() == "B")
    kitap_okumayi_seviyormusun = "bazen";
else if (dt.Rows[i]["s25"].ToString() == "C")
    kitap_okumayi_seviyormusun = "siksik";
else if (dt.Rows[i]["s25"].ToString() == "D")
    kitap_okumayi_seviyormusun = "herzaman";
else kitap_okumayi_seviyormusun = "bos";

//26
if (dt.Rows[i]["s26"].ToString() == "A")
    kac_saat_tv = "kullanmiyorum";
else if (dt.Rows[i]["s26"].ToString() == "B")
    kac_saat_tv = "1_2_saat";
else if (dt.Rows[i]["s26"].ToString() == "C")
    kac_saat_tv = "2_4_saat";
else if (dt.Rows[i]["s26"].ToString() == "D")
    kac_saat_tv = "4_saaten_fazla";
else kac_saat_tv = "bos";

//27
if (dt.Rows[i]["s27"].ToString() == "A")
    kac_saat_bilgisayar_kullanıyorsun = "kullanmiyorum";
else if (dt.Rows[i]["s27"].ToString() == "B")
    kac_saat_bilgisayar_kullanıyorsun = "1_2_saat";
else if (dt.Rows[i]["s27"].ToString() == "C")
    kac_saat_bilgisayar_kullanıyorsun = "2_4_saat";
else if (dt.Rows[i]["s27"].ToString() == "D")
    kac_saat_bilgisayar_kullanıyorsun = "4_saaten_fazla";
else kac_saat_bilgisayar_kullanıyorsun = "bos";

//28
if (dt.Rows[i]["s28"].ToString() == "A")
    evcil_hayvan= "evet";
else if (dt.Rows[i]["s28"].ToString() == "B")
    evcil_hayvan= "hayir";
else evcil_hayvan = "bos";

//29
if (dt.Rows[i]["s29"].ToString() == "A")
    sira_arakadasindan_memnunumusun = "memnundegilim";
else if (dt.Rows[i]["s29"].ToString() == "B")
    sira_arakadasindan_memnunumusun = "bazenmemnunum";
else if (dt.Rows[i]["s29"].ToString() == "C")
    sira_arakadasindan_memnunumusun = "geneldememnunum";
else if (dt.Rows[i]["s29"].ToString() == "D")
    sira_arakadasindan_memnunumusun = "cokmemnunum";
else sira_arakadasindan_memnunumusun = "bos";
/// 30
if (dt.Rows[i]["s30"].ToString() == "A")
    okul_etkinliginde_gorev = "hicbirzaman";
else if (dt.Rows[i]["s30"].ToString() == "B")
    okul_etkinliginde_gorev = "bazen";
else if (dt.Rows[i]["s30"].ToString() == "C")
    okul_etkinliginde_gorev = "siksik";
else if (dt.Rows[i]["s30"].ToString() == "D")
    okul_etkinliginde_gorev = "herzaman";
else okul_etkinliginde_gorev = "bos";
// 31
if (dt.Rows[i]["s31"].ToString() == "A")
    uyku_saati= "saat8de";
else if (dt.Rows[i]["s31"].ToString() == "B")
    uyku_saati= "saat9da";
else if (dt.Rows[i]["s31"].ToString() == "C")
    uyku_saati= "saat10da";
else if (dt.Rows[i]["s31"].ToString() == "D")
    uyku_saati = "saat11veyadahagec";
else uyku_saati= "bos";
//32
if (dt.Rows[i]["s32"].ToString() == "A")
    ogretmen_isimlerini_biliyormusun = "hicbirini_bilmiyorum";
```

```
else if (dt.Rows[i]["s32"].ToString() == "B")
    ogretmen_isimlerini_biliyormusun = "birkacini_biliyorum";
else if (dt.Rows[i]["s32"].ToString() == "C")
    ogretmen_isimlerini_biliyormusun = "cogunun_ismini_biliyorum";
else if (dt.Rows[i]["s32"].ToString() == "D")
    ogretmen_isimlerini_biliyormusun = "hepsinin_ismini_biliyorum";
else ogretmen_isimlerini_biliyormusun = "bos";

string degerx="";
if (dt_okul.Rows[0]["probleme_dayali"].ToString() == "True")
{
    degerx = "PDO";
}
else
{
    degerx = "GELENEKSEL";
}
string ekle = "insert into anket_aktar
(okul_id,ogrenci_no,probleme_dayali,ontest,sontest,gelir_duzeyi,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s
11,s12,s13,s14,s15,s16,s17,s18,s19,s20,s21,s22,s23,s24,s25,s26,s27,s28,s29,s30,s31,s32)
values(@okul_id,@ogrenci_no,@probleme_dayali,@ontest,@sontest,@gelir_duzeyi,@s1,@s2,@s3,@s4,@s5,
@s6,@s7,@s8,@s9,@s10,@s11,@s12,@s13,@s14,@s15,@s16,@s17,@s18,@s19,@s20,@s21,@s22,@s23,@s24,@s25,
@s26,@s27,@s28,@s29,@s30,@s31,@s32)";
OleDbCommand cmd = new OleDbCommand(ekle,con );
cmd.Parameters.AddWithValue("@okul_id",dt.Rows[i]["okul_id"]);
cmd.Parameters.AddWithValue("@ogrenci_no", dt.Rows[i]["ogrenci_no"]);
cmd.Parameters.AddWithValue("@probleme_dayali", degerx  );
cmd.Parameters.AddWithValue("@ontest", dt.Rows[i]["ontest"]);
cmd.Parameters.AddWithValue("@sontest", dt.Rows[i]["sontest"]);
cmd.Parameters.AddWithValue("@gelir_duzeyi", dt_okul.Rows[0]["gelir_duzeyi"]);
cmd.Parameters.AddWithValue("@s1", cinsiyet );
cmd.Parameters.AddWithValue("@s2", yas);
cmd.Parameters.AddWithValue("@s3", nerede_yasiyorsun );
cmd.Parameters.AddWithValue("@s4", kiminle_yasiyorsun);
cmd.Parameters.AddWithValue("@s5", odan_varmi);
cmd.Parameters.AddWithValue("@s6", annen_sagmi);
cmd.Parameters.AddWithValue("@s7", annen_calisiyormu);
cmd.Parameters.AddWithValue("@s8", annenin_egitim_durumu);
cmd.Parameters.AddWithValue("@s9", baban_sagmi);
cmd.Parameters.AddWithValue("@s10", baban_calisiyormu);
cmd.Parameters.AddWithValue("@s11", babanin_egitim_durumu);
cmd.Parameters.AddWithValue("@s12", kac_kardessiniz);
cmd.Parameters.AddWithValue("@s13", engelli_akraba);
cmd.Parameters.AddWithValue("@s14", odevlerine_kim_yardimci_oluyor);
cmd.Parameters.AddWithValue("@s15", telefon_numarasini_biliyormusun);
cmd.Parameters.AddWithValue("@s16", adresini_biliyormusun);
cmd.Parameters.AddWithValue("@s17", ulasim);
cmd.Parameters.AddWithValue("@s18", okulda_vakit_gecirme);
cmd.Parameters.AddWithValue("@s19", en_coksevdigin_ders);
cmd.Parameters.AddWithValue("@s20", ders_calisma_vakti);
cmd.Parameters.AddWithValue("@s21", odev_yapıyormusun);
cmd.Parameters.AddWithValue("@s22", arakadaslar_ile_vakit_gecirme);
cmd.Parameters.AddWithValue("@s23", yaz_tatili);
cmd.Parameters.AddWithValue("@s24", kis_tatili);
cmd.Parameters.AddWithValue("@s25", kitap_okumayi_seviyormusun);
cmd.Parameters.AddWithValue("@s26", kac_saat_bilgisayar_kullanıyorsun);
cmd.Parameters.AddWithValue("@s27", kac_saat_tv);
cmd.Parameters.AddWithValue("@s28", evcil_hayvan);
cmd.Parameters.AddWithValue("@s29", sira_arakadasindan_memnunumusun);
cmd.Parameters.AddWithValue("@s30", okul_etkinliginde_gorev);
cmd.Parameters.AddWithValue("@s31", uyku_saati);
cmd.Parameters.AddWithValue("@s32", ogretmen_isimlerini_biliyormusun);
cmd.ExecuteNonQuery();

cinsiyet = "";
yas = "";
nerede_yasiyorsun = "";
kiminle_yasiyorsun = "";
odan_varmi = "";
annen_sagmi = "";
annen_calisiyormu = "";
```

```
annenin_egitim_durumu = "";
baban_sagmi = "";
baban_calisiyormu = "";
babanin_egitim_durumu = "";
kac_kardessiniz = "";
engelli_akraba = "";
odevlerine_kim_yardimci_oluyor = "";
telefon_numarasini_biliyormusun = "";
adresini_biliyormusun = "";
ulasim = "";
okulda_vakit_gecirme = "";
en_coksevdigin_ders = "";
ders_calisma_vakti = "";
odev_yapıyormusun = "";
arakadaslar_ile_vakit_gecirme = "";
yaz_tatili = "";
kis_tatili = "";
kitap_okumayi_seviyormusun = "";
kac_saat_bilgisayar_kullanıyorsun = "";
kac_saat_tv = "";
evcil_hayvan = "";
sira_arakadasindan_memnunumusun = "";
okul_etkinliginde_gorev = "";
uyku_saati = "";
ogretmen_isimlerini_biliyormusun = "";
}
string anket_aktar_liste = "select * from anket_aktar ";

OleDbCommand cmdxy = new OleDbCommand(anket_aktar_liste, con);
OleDbDataAdapter dapy = new OleDbDataAdapter(cmdxy);
DataTable dty = new DataTable();
dapy.Fill(dty);
this.Text = dty.Rows.Count.ToString();

MessageBox.Show("Aktarma Tamamlandı");

con.Close();

}
```

Oluşturulan veritabanında yer alan öğrenci yanıtları incelendiğinde bazı cevaplarda mantık hataları olduğu tespit edilmiştir. Örneğin ***Annen yaşıyor mu?*** sorusuna Hayır cevabı vermesine rağmen **Kiminle yaşıyorsun?** sorusuna **Annemle**, **Evde derslerine yardımcı olanlar aşağıdakilerden hangisidir?** sorusuna yine **Annemle** cevabı verilen durumlar çok olmasa da tespit edilmiştir. Bu durumun **Baban yaşıyor mu?** sorusu için verilen cevaplar içinde incelenmiştir. Bu sorunu barındıran kayıtları veritabanından çıkarmak için **Anne ve Baba Sağ Olmama Durumuna Göre, Ders Çalışma ve Kalma Durumu** başlıklı bir komut düğmesi oluşturularak gerekli kodlar click() olayına yazılmıştır.

Aşağıda **Anne ve Baba Sağ Olmama Durumuna Göre, Ders Çalışma ve Kalma Durumu** komut düğmesi altında kullanılan kaynak kodlar verilmiştir.

```
private void button11_Click(object sender, EventArgs e)
{
```

```
OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=data.accdb");
con.Open();

string anket_aktar_cikti_sil = "delete * from anket_aktar where s6='hayir' and (s4='annemle' or
s13='annem')";
OleDbCommand cmd_anket_akter_cikti_sil = new OleDbCommand(anket_aktar_cikti_sil, con);
cmd_anket_akter_cikti_sil.ExecuteNonQuery();

string anket_aktar_cikti_sil1 = "delete * from anket_aktar where s9='hayir' and (s4='babamla' or
s13='babam')";
OleDbCommand cmd_anket_akter_cikti_sil1 = new OleDbCommand(anket_aktar_cikti_sil1, con);
cmd_anket_akter_cikti_sil1.ExecuteNonQuery();
con.Close();
}
```

Bir diğer mantık hatası veritabanında yer alan öntest ve sontest yanıtları incelendiğinde görülmektedir. Öntest ve sontest sonuçlarına bakıldığında beş öğrencinin öntestten aldığı başarı notunun 45 yüksek olmasına rağmen ünite bittikten sonra uygulanan başarı testi olan sontest sınavında aldığı başarı notu 45'in altında olduğu görülmektedir. Bu sorunun yaşandığı kayıtların veritabanından çıkarılması için **Öntest Başarılı Sontest Başarısız Olanları Sil** başlıklı komut düğmesi eklenerek gerekli kodlar eklenmiştir.

Aşağıda **Öntest Başarılı Sontest Başarısız Olanları Sil** komut düğmesi altında kullanılan kaynak kodlar verilmiştir.

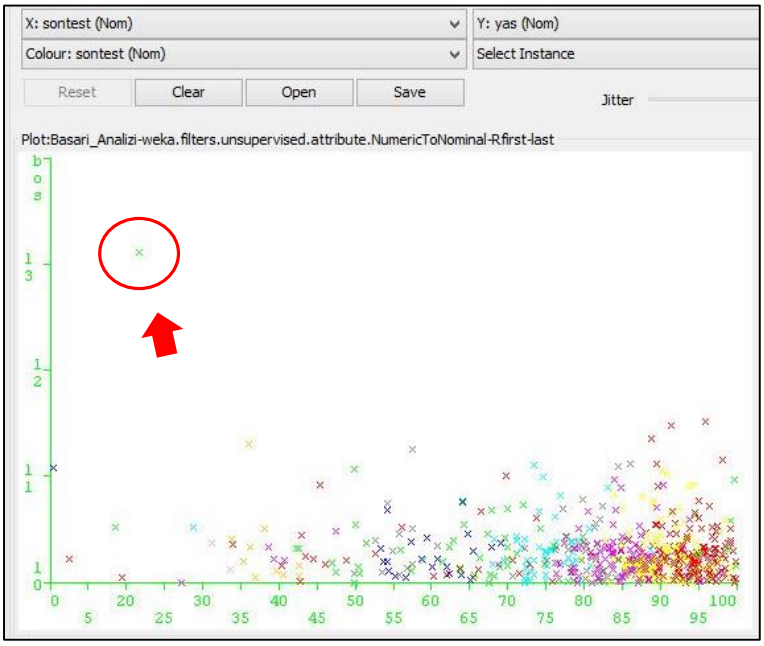
```
private void button7_Click_3(object sender, EventArgs e)
{
    OleDbConnection con = new OleDbConnection("Provider=Microsoft.Ace.Oledb.12.0;Data
Source=data.accdb");
    con.Open();

    string sil = "delete from anket_aktar where ontest>=45 and (sontest> 0 and sontest<45)";
    OleDbCommand cmd = new OleDbCommand(sil,con);
    cmd.ExecuteNonQuery();

    con.Close();
}
```

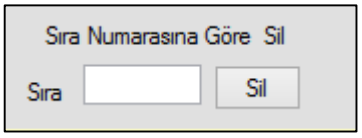
Oluşturulan veritabanında akla gelen ve gözle görülen mantık hataları barındıran örnekler yukardaki kodlar kullanılarak temizlendi. Bakıldığında görülebilen hatalar yanında bazen de gözle görülmeyecek anlamsız kayıtlar veritabanında barınabilir. Bunları tespit etmek için elde edilen veri kümesi üzerinde weka programının yeteneklerinden faydalanıldı. Weka programının kullanıcısına sunduğu bir ekran olan Visualize ekranı kullanılarak iki özniteliğin karşılıklı örnek dağılımı gözlemlenebilmektedir. Bu ekran kullanılarak tespit edilen anlamsız örnekler öğrenci satır numarası kullanılarak veritabanından çıkarılmıştır. Aşağıdaki resimde sontest ile yaş özniteliği örneklerinin

dağılımı görüntülenmiştir. Bu ekranda ilgili işaretçi çift tıklanarak örneğin sıra numarası öğrenilmektedir.



Şekil 3.5 Sontest özniteliği ile yaş özniteliği

Öğrenilen sıra numarası program arayüzünde bulunan **Sıra Numarasına Göre Sil** bölümüne girildikten sonra bu bölümde bulunan **Sil** komut düğmesi kullanılarak veri kümesinden silinmiştir.



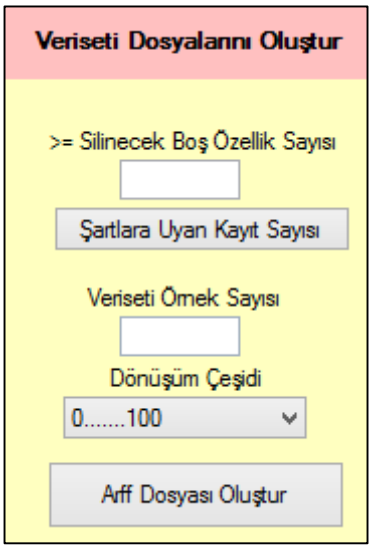
Şekil 3.6 Veritabanından Sil

Aşağıda **Sil** komut düğmesi altında kullanılan kaynak kodlar verilmiştir.

```
private void btn_sil_Click(object sender, EventArgs e)
{
DialogResult dr = MessageBox.Show("Kayıt Silinecek Onaylıyormusunuz?",
"Sil",MessageBoxButtons.YesNo,MessageBoxIcon.Warning );
OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=data.accdb ");
con.Open();
string anket_aktar = "select * from anket_aktar";
OleDbDataAdapter dap_anket_aktar = new OleDbDataAdapter(anket_aktar, con);
DataTable dt_anket_aktaR = new DataTable();
dap_anket_aktar.Fill(dt_anket_aktaR);
if (dr == DialogResult.Yes)
{
string sil = "delete from anket_aktar where id=@a ";
OleDbCommand cmd = new OleDbCommand(sil,con );
cmd.Parameters.AddWithValue("@a",dt_anket_aktaR.Rows[Convert.ToInt16 (txt_sira.Text
)]["id"].ToString ());
```

```
cmd.ExecuteNonQuery();
MessageBox.Show("Kayıt Silindi");
}
con.Close();
}
```

3.3.3. Veriseti Dosyalarını Oluştur



Şekil 3.7 Veriseti Dosyalarını Oluştur

Geliştirilen program arayüzünde tanıtımı yapılacak olan son bölüm **Veriseti dosyalarını oluştur** bölümüdür. Bu bölümde düzenlenebilecek üç alan bulunmaktadır. Bunlardan ilki **silinecek boş özellik sayısı** olarak isimlendiren alandır. Tez çalışması kapsamında oluşturulan veritabanında öğrenci numarası, anket, test vb.. sonuçları barındırmak için kullanılan 36 özniteliğe sahip bir tablo yer almaktadır. Bu tabloda bulunan alanlara bakıldığında bazı kayıtlarda özniteliklere değer girilmeyip boş geçildiği görülmektedir. Oluşturulacak veri kümesinde sonradan özniteliğin boş geçilmesinden kaynaklanacak sorunların yaşanmaması için özniteliği boş geçilen tüm kayıtların çıkarıldığı bir filtreleme işlemi **silinecek boş özellik sayısı** alanına 1 girilerek uygulanmıştır. Uygulama sonucunda koşulları karşılayan 570 örnek elde edilmiştir.

Tez çalışması kapsamında yapılacak doygunluk deneylerinde kullanılmak üzere farklı örnek sayısına sahip alt veri kümeleri kullanılması gerekmektedir. Farklı büyüklükteki alt veri kümeleri oluşturmak için bu bölümdeki **Veri Seti örnek sayısı** alanı kullanılmıştır. Burada yapılan alt küme işleminde oluşturulan alt veri kümelerindeki örneklerin her biri 570 örnekli ana veri kümesinden tekrarsız olarak seçilerek

oluşturulmuştur. Doygunluk deneylerinde kullanılan alt veri kümelerinin büyüklükleri 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 570 şeklindedir.

Eğitim veri kümesini kullanan pek çok çalışmada olduğu gibi bu çalışmada da kullanılan veri kümemizde sınıflandırma değeri başarı testi sonucunu barındıran (classify Value) sontest özniteliğidir. Bu özellik çalışmada öncelikle 0, 5, 10, …, 100'e toplam 21 sınıflı olarak kullanıldı. Daha sonra başarısız, geçer, orta, iyi, pekiyi olacak şekilde 5 sınıflı olarak ve en son başarılı, başarısız olacak şekilde 2 sınıflı. Çalışmada kullanılan bu farklı sayıda sınıflandırma değeri oluşturulması işlemi **Veriseti dosyalarını oluştur** bölümündeki **Dönüşüm Çeşidi** açılır menüsü seçenekleri kullanılarak belirlendi.

Tez çalışmasında veri kümesinde yapılan deneyler bir makine öğrenmesi aracı olan Weka programı kullanılarak gerçekleştirilmiştir. Bu bölüme özel ayarlamalar yapıldıktan sonra gerçekleştirilen filtreleme işlemlerinden elde edilen veri kümesinin Weka programının anlayacağı biçime dönüştürülmesi gerekmektedir. Weka programı .arff, .csv vb.. biçimdeki dosyaları kullanabilmektedir. Bu çalışmada filtreleme sonrasında elde edilen veriler .arff formatında bir dosyaya kaydedilmiştir. Bu işlemi yapan kodlar **Arff Dosyası Oluştur** komut düğmesinin click() olayına yazılmıştır.

Aşağıda **Arff Dosyası Oluştur** komut düğmesi altında kullanılan kaynak kodlar verilmiştir.

```
private void btn_dosya_oluştur_Click(object sender, EventArgs e)
{
if (txt_bos_ozellik.Text == "")
{
    txt_bos_ozellik.Text = "100";
}
OleDbConnection con=new OleDbConnection ("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=data.accdb");
con.Open ();

string anket_aktar_cikti_sil = "delete * from anket_aktar_cikti";
OleDbCommand cmd_anket_akter_cikti_sil = new OleDbCommand(anket_aktar_cikti_sil,con );
cmd_anket_akter_cikti_sil.ExecuteNonQuery();

string anket_aktar = "select * from anket_aktar ";
OleDbDataAdapter dap_kaynak=new OleDbDataAdapter (anket_aktar ,con );
DataTable dt_kaynak=new DataTable ();
dap_kaynak.Fill (dt_kaynak);

string anket_aktar_cikti_listele = "select * from anket_aktar_cikti";
OleDbDataAdapter dap_anket_aktar_cikti = new OleDbDataAdapter(anket_aktar_cikti_listele,con );
DataTable dt = new DataTable();
dap_anket_aktar_cikti.Fill(dt);
int toplam = 0;
for (int i = 0; i < dt_kaynak.Rows.Count; i++)
{
toplam = 0;

for (int k = 1; k < dt_kaynak.Columns.Count; k++)
{
    if (dt_kaynak.Rows[i][k].ToString() == "bos")
```

```
        toplam = toplam + 1;
}
/// girilen değerden küçük ise ekle
if (toplam < Convert.ToInt32(txt_bos_ozellik.Text))
{
    dt.ImportRow(dt_kaynak.Rows[i]);
}
}
DataTable dt_clone = dt.Clone();
dt_clone.Clear();

this.Text = dt.Rows.Count.ToString();

//Int32 kalan_deger = dt_basarisiz .Rows.Count;
Int32 kalan_deger = 0;

if (txtsayi .Text != "")
{
if(dt.Rows.Count+kalan_deger  >=Convert.ToInt32 (txtsayi.Text))
{
    Random rnd = new Random();

    List<Int32> lst_sira_id = new List<Int32>();
    lst_sira_id.Clear();
    int k=0;// rnd.Next(1, dt.Rows.Count);

for (int i = 0; i < Convert.ToInt16(txtsayi.Text)-kalan_deger ; i++)
{

    Boolean varmi = true   ;

        if (lst_sira_id.Count+kalan_deger  < (Convert.ToInt16(txtsayi.Text) ))
        {
            do   ///// DAHA ÖNCE ALINMAYAN KAYITLAR ALINSIN
            {
               k = rnd.Next(0, dt.Rows.Count);

                if (lst_sira_id.Count > 0)
                {
                    if (lst_sira_id.Contains(k))
                    {
                        varmi = true;
                    }
                    else
                    {
                        lst_sira_id.Add(k);
                        varmi = false;
                  }
                }
                else
                {
                    lst_sira_id.Add(k);
                    varmi = false;
                }
            } while (varmi == true);
        }
    ////////////////////////////////////
    try
    {
        DataRow dr = dt_clone.NewRow();

        dr["okul_id"] = dt.Rows[k]["okul_id"];
        dr["ogrenci_no"] = dt.Rows[k]["ogrenci_no"];
        dr["probleme_dayali"] = dt.Rows[k]["probleme_dayali"].ToString();
        dr["ontest"] = dt.Rows[k]["ontest"].ToString();
        dr["sontest"] = dt.Rows[k]["sontest"].ToString();
        // dr["sontest"] = son_test_deger ;
        dr["gelir_duzeyi"] = dt.Rows[k]["gelir_duzeyi"].ToString();
        dr["s1"] = dt.Rows[k]["s1"].ToString();
        dr["s2"] = dt.Rows[k]["s2"].ToString();
        dr["s3"] = dt.Rows[k]["s3"].ToString();
```

```
        dr["s4"] = dt.Rows[k]["s4"].ToString();
        dr["s5"] = dt.Rows[k]["s5"].ToString();
        dr["s6"] = dt.Rows[k]["s6"].ToString();
        dr["s7"] = dt.Rows[k]["s7"].ToString();
        dr["s8"] = dt.Rows[k]["s8"].ToString();
        dr["s9"] = dt.Rows[k]["s9"].ToString();
        dr["s10"] = dt.Rows[k]["s10"].ToString();
        dr["s11"] = dt.Rows[k]["s11"].ToString();
        dr["s12"] = dt.Rows[k]["s12"].ToString();
        dr["s13"] = dt.Rows[k]["s13"].ToString();
        dr["s14"] = dt.Rows[k]["s14"].ToString();
        dr["s15"] = dt.Rows[k]["s15"].ToString();
        dr["s16"] = dt.Rows[k]["s16"].ToString();
        dr["s17"] = dt.Rows[k]["s17"].ToString();
        dr["s18"] = dt.Rows[k]["s18"].ToString();
        dr["s19"] = dt.Rows[k]["s19"].ToString();
        dr["s20"] = dt.Rows[k]["s20"].ToString();
        dr["s21"] = dt.Rows[k]["s21"].ToString();
        dr["s22"] = dt.Rows[k]["s22"].ToString();
        dr["s23"] = dt.Rows[k]["s23"].ToString();
        dr["s24"] = dt.Rows[k]["s24"].ToString();
        dr["s25"] = dt.Rows[k]["s25"].ToString();
        dr["s26"] = dt.Rows[k]["s26"].ToString();
        dr["s27"] = dt.Rows[k]["s27"].ToString();
        dr["s28"] = dt.Rows[k]["s28"].ToString();
        dr["s29"] = dt.Rows[k]["s29"].ToString();
        dr["s30"] = dt.Rows[k]["s30"].ToString();
        dr["s31"] = dt.Rows[k]["s31"].ToString();
        dr["s32"] = dt.Rows[k]["s32"].ToString();
        dt_clone.Rows.Add(dr);
    }
        catch (Exception sorun)
        {
            MessageBox.Show(sorun .ToString ());
        }
}

StreamWriter sw = new StreamWriter(Application.StartupPath + "\\" + txtsayi.Text +
DateTime.Now.Hour.ToString() + DateTime.Now.Minute.ToString() + ".arff", true);
sw.WriteLine("@Relation Basari_Analizi");
sw.WriteLine("");
sw.WriteLine("@Attribute probleme_dayali {PDO,GELENEKSEL,bos}");
sw.WriteLine("@Attribute gelir_duzeyi {kotu,orta,iyi,cok_iyi,bos}");
sw.WriteLine("@Attribute cinsiyet {kiz,erkek,bos}");
sw.WriteLine("@Attribute yas {10,11,12,13,bos}");
sw.WriteLine("@Attribute nerede_yasiyorsun {ailemle,yurtta,akrabada,hicbiri,bos}");
sw.WriteLine("@Attribute kiminle_yasiyorsun {annemle,babamla,ailemle,diger,bos}");
sw.WriteLine("@Attribute odan_varmi{evet,hayir,bos}");
sw.WriteLine("@Attribute annen_yasiyormu{evet,hayir,bos}");
sw.WriteLine("@Attribute annen_calisiyormu{evet,hayir,bos}");
sw.WriteLine("@Attribute annenin_egitim_durumu
{ilkokul,ortaokul,lise,universite,Okuma_yazma_bilmiyor,bos }");
sw.WriteLine("@Attribute baban_yasiyormu{evet,hayir,bos}");
sw.WriteLine("@Attribute baban_calisiyormu{evet,hayir,bos}");
sw.WriteLine("@Attribute babanin_egitim_durumu
{ilkokul,ortaokul,lise,universite,Okuma_yazma_bilmiyor,bos }");
sw.WriteLine("@Attribute kac_kardessiniz{tekim,iki,uc,dort_ve_daha_fazla,bos}");
sw.WriteLine("@Attribute engelli_akraba {annem,babam,kardesim,hicbiri,bos}");
sw.WriteLine("@Attribute odevlerine_kim_yardimci_oluyor
{annem,babam,anne_ve_babam,diger,bos}");
sw.WriteLine("@Attribute telefon_numarasini_biliyormusun{evet,hayir,bos}");
sw.WriteLine("@Attribute adresini_biliyormusun {evet,hayir,bos}");
sw.WriteLine("@Attribute ulasim   {servis,araba,yuruyerek,minibus,bos}");
sw.WriteLine("@Attribute okulda_vakit_gecirme  {hicbirzaman,bazen,siksik,herzaman,bos}");
sw.WriteLine("@Attribute en_coksevdigin_ders   {fenveteknoloji,matematik,turkce,hicbiri,bos}");
sw.WriteLine("@Attribute ders_calisma_vakti  {hicbirzaman,bazen,siksik,herzaman,bos}");
sw.WriteLine("@Attribute odev_yapıyormusun   {hicbirzaman,bazen,siksik,herzaman,bos}");
sw.WriteLine("@Attribute arakadaslar_ile_vakit_gecirme
{hicbirzaman,bazen,siksik,herzaman,bos}");
sw.WriteLine("@Attribute yaz_tatili  {hicbirzaman,bazen,siksik,herzaman,bos}");
sw.WriteLine("@Attribute kis_tatili {hicbirzaman,bazen,siksik,herzaman,bos}");
```

```
sw.WriteLine("@Attribute kitap_okumayi_seviyormusun  {hicbirzaman,bazen,siksik,herzaman,bos}");
sw.WriteLine("@Attribute kac_saat_bilgisayar_kullanıyorsun
{kullanmiyorum,1_2_saat,2_4_saat,4_saaten_fazla,bos}");
sw.WriteLine("@Attribute kac_saat_tv    {kullanmiyorum,1_2_saat,2_4_saat,4_saaten_fazla,bos}");
sw.WriteLine("@Attribute evcil_hayvan  {evet,hayir,bos}");
sw.WriteLine("@Attribute sira_arakadasindan_memnunumusun
{memnundegilim,bazenmemnunum,geneldememnunum,cokmemnunum,bos}");
sw.WriteLine("@Attribute okul_etkinliginde_gorev   {hicbirzaman,bazen,siksik,herzaman,bos}");
sw.WriteLine("@Attribute uyku_saati    {saat8de,saat9da,saat10da,saat11veyadahagec,bos}");
sw.WriteLine("@Attribute ogretmen_isimlerini_biliyormusun
{hicbirini_bilmiyorum,birkacini_biliyorum,cogunun_ismini_biliyorum,hepsinin_ismini_biliyorum,bos
}");

if (cmb_donusum.SelectedIndex == 0)
{
    sw.WriteLine("@Attribute ontest
{0,5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100}");
    sw.WriteLine("@Attribute sontest
{0,5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100}");
}
else if (cmb_donusum.SelectedIndex == 1)
{
    sw.WriteLine("@Attribute ontest {BASARISIZ,GECER,ORTA,IYI,PEKIYI}");
    sw.WriteLine("@Attribute sontest {BASARISIZ,GECER,ORTA,IYI,PEKIYI}");
}
else if (cmb_donusum.SelectedIndex == 2)
{
    sw.WriteLine("@Attribute ontest {BASARILI,BASARISIZ}");
    sw.WriteLine("@Attribute sontest {BASARILI,BASARISIZ}");
}
sw.WriteLine("");

sw.WriteLine("@Data");
sw.WriteLine("");
string satir = "";
string on_test_deger = "";
string son_test_deger = "";

for (int i = 0; i < dt_clone.Rows.Count; i++)
{
    for (int j = 3; j < 37; j++)
    {
        satir = satir + dt_clone.Rows[i][j].ToString();
        if (j < 38)
        {
            satir = satir + ",";
        }
    }
    if (cmb_donusum.SelectedIndex == 2)
    {
        if (Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) <= 100 &&
Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) >= 64)
        {
            on_test_deger = "BASARILI";
        }
        else
        {
            on_test_deger = "BASARISIZ";
        }
        satir = satir + on_test_deger + ",";

        if (Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) <= 100 &&
Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) >= 64)
        {
         son_test_deger = "BASARILI";
        }
        else
        {
            son_test_deger = "BASARISIZ";
        }
```

```
        satir = satir + son_test_deger;
    }
    else if (cmb_donusum.SelectedIndex == 1)
    {
        if (Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) <= 100 &&
Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) >= 85)
        {
            on_test_deger = "PEKIYI";
        }
        else if (Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) <= 84 &&
Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) >= 70)
        {
            on_test_deger = "IYI";
        }
        else if (Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) <= 69 &&
Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) >= 55)
        {
            on_test_deger = "ORTA";
        }
        else if (Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) <= 54 &&
Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) >= 45)
        {
            on_test_deger = "GECER";
        }
        else if (Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) <= 44 &&
Convert.ToInt32(dt_clone.Rows[i]["ontest"].ToString()) >= 0)
        {
            on_test_deger = "BASARISIZ";
        }

        satir = satir + on_test_deger + ",";

        if (Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) <= 100 &&
Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) >= 85)
        {
            son_test_deger = "PEKIYI";
        }
        else if (Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) <= 84 &&
Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) >= 70)
        {
            son_test_deger = "IYI";
        }
        else if (Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) <= 69 &&
Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) >= 55)
        {
            son_test_deger = "ORTA";
        }
        else if (Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) <= 54 &&
Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) >= 45)
        {
            son_test_deger = "GECER";
        }
        else if (Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) <= 44 &&
Convert.ToInt32(dt_clone.Rows[i]["sontest"].ToString()) >= 0)
        {
            son_test_deger = "BASARISIZ";
        }

        satir = satir + son_test_deger;
    }
    else if (cmb_donusum.SelectedIndex == 0)
    {
        on_test_deger=(dt_clone.Rows[i]["ontest"].ToString());
        satir = satir +on_test_deger   + ",";
        son_test_deger = (dt_clone.Rows[i]["sontest"].ToString());
        satir = satir + son_test_deger;
    }
    ///////////////
    sw.WriteLine(satir);
    satir = "";
}
```

```
sw.Close();

lst_sira_id.Clear();

MessageBox.Show(txtsayi.Text + " isimli dosya oluşturuldu");
}
else
{
MessageBox.Show ("Girdiğiniz Değer Kadar Tekrarsız Sayıda Örnek Türetilemez. Max örnek
Sayisi="+dt.Rows.Count.ToString ());
}
}
else
{
txtsayi.Focus();
MessageBox.Show("Lütfen Seçim Yapın");
}
con.Close();
}
```

BÖLÜM 4

DENEYLER VE ANALİZ

4.1. Test ve anket sonuçlarının dijital ortama aktarılıp saklanması

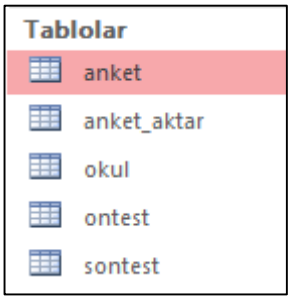
Yapılan çalışmada öğrencilerin anket ve test sorularına verdikleri yanıtların toplanmasında optik formlardan faydalanılmıştır. Verilerin alınmasında optik form kullanılarak verilerin doğru ve tam olarak toplanıp aktarılması, bunun yanında zamandan da tasarruf edilmesi sağlanmıştır.

Optik formların okunmasında Sekonik marka optik okuma cihazı kullanılmıştır. Cihaz okuduğu formların sonuçlarını kurulum klasöründe oluşturduğu yeni bir. dat dosyası içine yazıp saklamaktadır.

Oluşturulan dosyalara erişmek için veya veritabanlarına ulaşıp işlem yapmak için Borland, Microsoft vb. firmaların geliştirdiği editörler kullanılarak geliştirilen özel programlar kullanılabilmektedir. Bu çalışmada Visual Studio 2010 C# programı kullanarak geliştirilen yazılım ile optik okuyucu tarafından oluşturulan bilgi yüklü dosyalara erişilerek içindeki veriler okunmuştur.

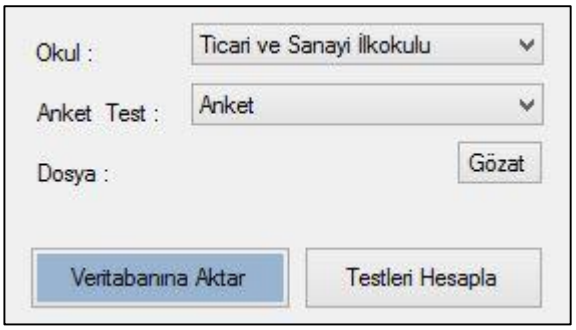
Dijital ortamda verileri düzenli bir şekilde kaydetmek, saklamak ve ihtiyaç halinde hızlıca erişebilmek için veritabanı programlarından faydalanılmaktadır. Birçok veritabanı programı olmakla birlikte bu çalışmada .dat uzantılı dosyalardan okunan bilgileri kaydetmek için Microsoft tarafından geliştirilen ve Microsoft Office 2013 paketinde yer alan Access veritabanı programı kullanılmıştır. Tüm veritabanı programlarında olduğu gibi Microsoft Office 2013 Access programında da öncelikle verilerin saklı tutulacağı veritabanı dosyasının oluşturulması gerekmektedir. Bu çalışmada data.accdb isimli bir veritabanı dosyası oluşturulmuştur. Daha sonra verilerin saklı tutulacağı tabloların oluşturulması gerekmektedir. Çalışmada kullanılan tablolar, okul bilgilerini saklamak için kullanılan **okul** isimli tablo, öntest verilerini saklamak için

öntest isimli tablo, sontest bilgilerini saklamak için **sontest** isimli tablo, anket, öntest, sontest bilgilerinin tamamının gürültüden temizlenmiş kopyasının tutulması için **anket_aktar** isimli tablodur.



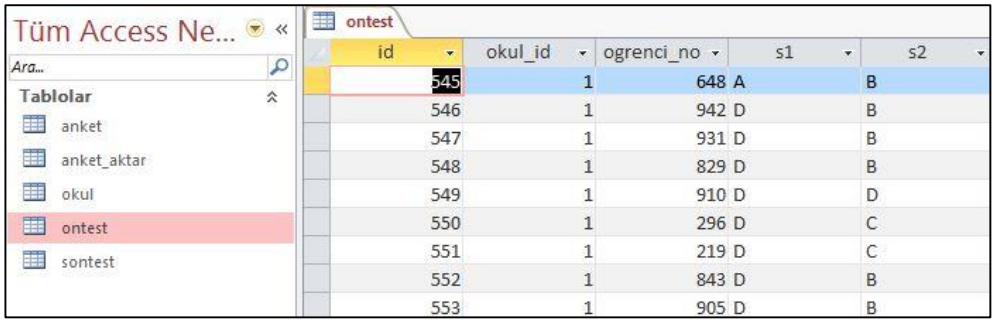
Şekil 4.1 Data isimli Veritabanında Kullanılan Tabloları

Çalışmada geliştirilen programa okuma eylemi için bir koşul eklenmiştir. Bu koşula göre eğer öğrenci, numara alanını boş veya hatalı doldurmuş ise programın bu satırı kaydetmeden bir sonraki satıra geçmesi sağlanmıştır. Aşağıdaki resimde. dat uzantılı dosyalardan veri alınmasında kullanılan program penceresi yer almaktadır. Bu pencerede bulunan okul alanında hangi okulun bilgilerinin okutulacağı belirlenmektedir. Anket – Test alanında ilgili okulun hangi bilgilerinin okunacağı seçilmektedir. Dosya bölümünde optik okuyucunun oluşturduğu dosyanın yeri seçilmektedir. Veritabanına Aktar düğmesine basılarak okunacak dosyanın sahip olduğu verilerin, koşulu sağlaması durumunda bilginin türüne göre anket ise anket, öntest ise öntest, sontest ise sontest tablosuna kaydetme işlemi gerçekleştirilmektedir. Testleri hesapla düğmesine basılarak öntest ve sontest tablosunda yer alan öğrenci cevapları hesaplanmaktadır.



Şekil 4.2 Veritabanına Ekle ve Hesapla

Aşağıdaki resimde öntest tablosuna eklenen kayıtlar gösterilmiştir.



Şekil 4.3 ontest Tablosu Kayıtları

4.2. 21 Sınıflı Sontest Özniteliğine Sahip Veri Kümesinin Analizi

Bir algoritmanın öğrenme sürecinin sonucunda ne kadar başarılı olduğuna karar verilmesi için çeşitli ölçütlere gereksinim vardır. En temel model başarım ölçütleri; doğruluk oranı, kesinlik, duyarlılık ve f ölçütüdür. Ayrıca modelin başarısı ölçülürken, yukarıda verilen ölçütlerin dışında, kappa istatistiği de ölçüt olarak kullanılabilir [47].

Bu bağlamda veri kaynağı 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 570 örneğe sahip toplam 12 alt veri kümesine ayrılmıştır. Her bir alt veri kümesi için Weka'nın sahip olduğu sınıflandırma algoritmaları çalıştırılmıştır. Elde edilen sonuçlarda tüm alt veri kümeleri için kappa değerinin çalışmada esas alınan 0,4 değerinin altında olduğu tespit edilmiştir. Bu durum kullanılan alt veri kümelerinden elde edilen başarının rastlantısal olduğunu ve veri kümelerinin dengesiz olduğunu göstermektedir. Bununla birlikte elde edilen F-measure değerinin de 0,5 değerinin altında olduğu görülmüştür. Bu sonuç bize oluşturulacak modellerin tanı için yeterli olmadığını göstermektedir. Aşağıda Naïve Bayes ve J48 sınıflandırma algoritmasının 570 örneğe sahip veri kümesi üzerinde çalıştırılması sonucunda elde edilen sonuçlar gösterilmiştir.

```
Correctly Classified Instances         142               24.9123 %
Kappa statistic                          0.0583
Mean absolute error                      0.0736
Root mean squared error                  0.212
Relative absolute error                 91.5068 %
Root relative squared error            105.9698 %
Coverage of cases (0.95 level)          72.2807 %
Mean rel. region size (0.95 level)      19.3567 %
Total Number of Instances              570

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000      0,000    0,000      0,000    0,993     0,200     0
               0,000    0,000    0,000      0,000    0,000      0,000    0,978     0,105     5
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         10
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         15
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         20
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         25
               0,000    0,000    0,000      0,000    0,000      0,000    0,884     0,022     30
               0,000    0,000    0,000      0,000    0,000      0,000    0,861     0,044     35
               0,000    0,002    0,000      0,000    0,000      -0,002   0,993     0,267     40
               0,000    0,002    0,000      0,000    0,000      -0,006   0,782     0,066     45
               0,000    0,007    0,000      0,000    0,000      -0,013   0,709     0,044     50
               0,000    0,009    0,000      0,000    0,000      -0,014   0,785     0,062     55
               0,190    0,018    0,286      0,190    0,229      0,210    0,815     0,158     60
               0,077    0,004    0,333      0,077    0,125      0,151    0,687     0,084     65
               0,000    0,011    0,000      0,000    0,000      -0,022   0,529     0,054     70
               0,111    0,076    0,111      0,111    0,111      0,035    0,557     0,113     75
               0,021    0,044    0,042      0,021    0,028      -0,032   0,636     0,116     80
               0,124    0,135    0,145      0,124    0,133      -0,012   0,507     0,161     85
               0,242    0,224    0,231      0,242    0,236      0,017    0,576     0,256     90
               0,600    0,407    0,345      0,600    0,438      0,170    0,654     0,368     95
               0,000    0,000    0,000      0,000    0,000      0,000    0,521     0,014     100
Weighted Avg.  0,249    0,188    0,194      0,249    0,210      0,056    0,617     0,212
```

Şekil 4.4 21 Sınıflı Veri Kümesinin Naïve Bayes Algoritması Sonuçları

Tablo 4.1 Naïve Bayes Sonucu

Algoritma	**Instances**	**Correctly Classified**	**Kappa**	**Precision**	**Recall**	**F-Meausre**
Naïve Bayes	570	0,25	0,058	0,19	0,25	0,21

```
Correctly Classified Instances          96               16.8421 %
Kappa statistic                          0.0163
Mean absolute error                      0.0784
Root mean squared error                  0.2476
Relative absolute error                 97.4427 %
Root relative squared error            123.7561 %
Coverage of cases (0.95 level)          37.5439 %
Mean rel. region size (0.95 level)      10.8939 %
Total Number of Instances              570

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,000    0,000    0,000      0,000    0,000      0,000    0,497     0,002     0
                 0,000    0,005    0,000      0,000    0,000      -0,004   0,495     0,004     5
                 0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         10
                 0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         15
                 0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         20
                 0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         25
                 0,000    0,000    0,000      0,000    0,000      0,000    0,996     0,417     30
                 0,000    0,007    0,000      0,000    0,000      -0,008   0,489     0,009     35
                 0,000    0,002    0,000      0,000    0,000      -0,002   0,496     0,004     40
                 0,000    0,014    0,000      0,000    0,000      -0,016   0,485     0,018     45
                 0,143    0,025    0,125      0,143    0,133      0,110    0,539     0,078     50
                 0,077    0,034    0,050      0,077    0,061      0,035    0,544     0,030     55
                 0,048    0,047    0,037      0,048    0,042      0,000    0,507     0,038     60
                 0,077    0,020    0,083      0,077    0,080      0,059    0,494     0,029     65
                 0,000    0,040    0,000      0,000    0,000      -0,043   0,488     0,043     70
                 0,178    0,076    0,167      0,178    0,172      0,099    0,568     0,124     75
                 0,104    0,105    0,083      0,104    0,093      -0,001   0,521     0,097     80
                 0,135    0,152    0,141      0,135    0,138      -0,017   0,509     0,161     85
                 0,202    0,200    0,219      0,202    0,210      0,002    0,538     0,230     90
                 0,273    0,245    0,285      0,273    0,279      0,028    0,517     0,262     95
                 0,000    0,011    0,000      0,000    0,000      -0,011   0,475     0,011     100
Weighted Avg.    0,168    0,152    0,172      0,168    0,170      0,015    0,524     0,171
```

Şekil 4.5 J48 Algoritması Sonucu

Tablo 4.2 21 Sınıflı Veri Kümesinin J48 Algoritması Sonuçları

Algoritma	**Instances**	**Correctly Classified**	**Kappa**	**Precision**	**Recall**	**F-Meausre**
J48	570	0,17	0,016	0,17	0,17	0,17

4.3. 5 Sınıflı Sontest Özniteliğine Sahip Veri Kümesinin Analizi

Bir algoritmanın öğrenme sürecinin sonucunda ne kadar başarılı olduğuna karar verilmesi için çeşitli ölçütlere gereksinim vardır. En temel model başarım ölçütleri; doğruluk oranı, kesinlik, duyarlılık ve f ölçütüdür. Ayrıca modelin başarısı ölçülürken, yukarıda verilen ölçütlerin dışında, kappa istatistiği de ölçüt olarak kullanılabilir [47].

Bu bağlamda veri kaynağı 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 570 örneğe sahip toplam 12 alt veri kümesine ayrılmıştır. Her bir alt veri kümesi için Weka'nın sahip olduğu sınıflandırma algoritmaları çalıştırılmıştır. Sonuçlarda tüm alt veri kümeleri için kappa değerinin çalışmada esas alınan 0,4 değerinin altında olduğu tespit edilmiştir. Bu durum kullanılan alt veri kümelerinden elde edilen başarının rastlantısal olduğunu ve veri kümelerinin dengesiz olduğunu göstermektedir. Bununla birlikte elde edilen F-measure değerinin de literatürden belirlenen 0,5 değerinin altında veya çok az üstünde olduğu görülmüştür. Bu değerin tek başına istenen sınırlarda olması yeterli olmayacağından sınıf sayısı azaltılarak denemelere devam edilmiştir. Aşağıda Naïve Bayes ve J48 sınıflandırma algoritmasının 570 örneğe sahip veri kümesi üzerinde çalıştırılması sonucunda elde edilen sonuçlar gösterilmiştir.

```
Correctly Classified Instances         355               62.2807 %
Kappa statistic                          0.2008
Mean absolute error                      0.1641
Root mean squared error                  0.3244
Relative absolute error                 77.1485 %
Root relative squared error             99.72   %
Coverage of cases (0.95 level)          90.8772 %
Mean rel. region size (0.95 level)      41.2281 %
Total Number of Instances              570

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,167     0,005     0,400       0,167   0,235      0,248    0,952     0,393     BASARISIZ
               0,000     0,016     0,000       0,000   0,000      -0,027   0,829     0,131     GECER
               0,468     0,059     0,415       0,468   0,440      0,387    0,847     0,423     ORTA
               0,102     0,122     0,179       0,102   0,130      -0,025   0,603     0,243     IYI
               0,864     0,582     0,732       0,864   0,793      0,318    0,739     0,814     PEKIYI
Weighted Avg.  0,623     0,408     0,553       0,623   0,581      0,237    0,728     0,626
```

Şekil 4.6 Naïve Bayes Algoritması Weka Programı Çıktısı

Tablo 4.3 Naïve Bayes Algoritması Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Meausre
Naïve Bayes	570	0,62	0,20	0,55	0,62	0,58

```
Correctly Classified Instances         379               66.4912 %
Kappa statistic                          0.2487
Mean absolute error                      0.1742
Root mean squared error                  0.3204
Relative absolute error                 81.9318 %
Root relative squared error             98.5026 %
Coverage of cases (0.95 level)          89.1228 %
Mean rel. region size (0.95 level)      44.807  %
Total Number of Instances              570

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,167     0,018     0,167       0,167   0,167      0,149    0,737     0,191     BASARISIZ
                 0,250     0,031     0,261       0,250   0,255      0,223    0,528     0,209     GECER
                 0,277     0,044     0,361       0,277   0,313      0,263    0,520     0,228     ORTA
                 0,059     0,042     0,269       0,059   0,097      0,034    0,576     0,246     IYI
                 0,951     0,607     0,742       0,951   0,834      0,438    0,706     0,764     PEKIYI
Weighted Avg.    0,665     0,407     0,580       0,665   0,600      0,325    0,657     0,577
```

Şekil 4.7 J48 Algoritması Weka Programı Çıktısı

Tablo 4.4 J48 Algoritması Sonuçları

Algoritma	**Instances**	**Correctly Classified**	**Kappa**	**Precision**	**Recall**	**F-Meausre**
J48	570	0,67	0,25	0,58	0,67	0,60

4.4. 2 Sınıflı Sontest Özniteliğine Sahip Veri Kümesi Doygunluk Deneyleri

Bilindiği gibi sınıflandırma algoritmalarının öğrenme sürecinde veri kümelerinde yer alan örnekler üzerinden bilgi sağlanır ve algoritmalar bu bilgi ile eğitilir. Örnekler üzerinden elde edilen bilginin yeterli veya yetersiz olması sınıflandırma başarısını doğrudan etkilemektedir. Çünkü yeterli miktarda örnek ile örneklenmemiş verilerin sınıflandırması büyük oranda şansa bağlı olacaktır. Örnek miktarının sınıflandırıcı performansını nasıl etkilediğini anlamak için veri kümesinden oransal olarak artan sayıda alt veri kümeleri seçilerek deneyler yapılmalı ve deneylerden elde edilen örnek büyüklüğü ile f-ölçütü değerleri arasındaki ilişkiyi gösteren mutluluk grafikleri (happy graphs) oluşturularak incelenmelidir.

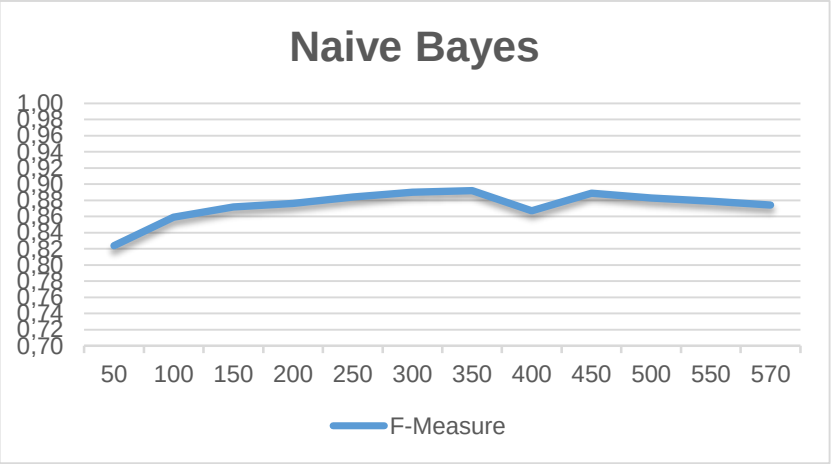
Çalışmada veri kümesinin sahip olduğu örnekler, ilki 50 örneğe sahip olacak şekilde sonrakileri 50 şer arttırılarak 100 örnekli, 150 örnekli ….550 örnekli şeklinde artan ve en son 570 örnekten oluşan alt veri kümeleri seçilmiştir.

Weka 3.7. programının classify sekmesinde bulunan sınıflandırma algoritmaları alt veri kümelerinin analiz edilmesinde kullanılmıştır. Elde edilen sonuçlar tablolarla ve örnek büyüklüğü ile f-ölçütü değeri ilişkisini gösteren mutluluk grafikleri ile sunulmuştur. Tablolar ve grafikler incelenmiş ve yorumlanmıştır.

4.4.1. Naïve Bayes ile Doygunluk Deneyleri

Tablo 4.5 Naïve Bayes Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
bayes.Naïve Bayes	50	0,88	0,00	0,77	0,88	0,82	0,34
bayes.Naïve Bayes	100	0,88	0,35	0,86	0,88	0,86	0,34
bayes.Naïve Bayes	150	0,89	0,26	0,87	0,89	0,87	0,31
bayes.Naïve Bayes	200	0,88	0,58	0,87	0,88	0,88	0,32
bayes.Naïve Bayes	250	0,89	0,43	0,88	0,89	0,88	0,31
bayes.Naïve Bayes	300	0,89	0,52	0,89	0,89	0,89	0,31
bayes.Naïve Bayes	350	0,89	0,50	0,89	0,89	0,89	0,30
bayes.Naïve Bayes	400	0,87	0,47	0,87	0,87	0,87	0,33
bayes.Naïve Bayes	450	0,89	0,43	0,89	0,89	0,89	0,30
bayes.Naïve Bayes	500	0,89	0,51	0,88	0,89	0,88	0,31
bayes.Naïve Bayes	550	0,88	0,46	0,88	0,88	0,88	0,31
bayes.Naïve Bayes	570	0,88	0,49	0,87	0,88	0,87	0,32



Şekil 4.8 Naïve Bayes Algoritması Mutluluk Grafiği

570 örneğe sahip veri kümesinden, oransal olarak giderek büyüyen en az 50 örneğe sahip olmak üzere 100, 150, 200, 250, 300, 350, 400, 450, 500 ve 570 örneğe sahip olacak şekilde alt veri kümeleri seçilmiştir. Daha sonra seçilen alt veri kümeleri üzerinde Naïve Bayes sınıflandırma algoritması kullanılarak deneyler yapılmıştır.

Deney sonuçlarına bakıldığında performansı en yüksek veri kümesinin 300 örneğe sahip alt veri kümesi olduğu görülmektedir. Bu sonuçlara göre veri kümesinin doğru sınıflandırma başarısının %89 olduğu görülmektedir. Alt veri kümeler ile yapılan sınıflandırma çalışması sonuçlarının tamamına bakıldığında performans sonuçlarının çok yakın olduğu görülmektedir. Bu eğitim verisinin yeterli olup olmadığı konusunda fikir vermektedir. Şöyle ki eğitim verisinin miktarının arttırılmasının, sınıflandırma algoritmasının performansında bir değişiklik yaratmadığını veya performansının yakın olması verilerin sayıca yeterli olduğuna ilişkin bir gösterge olarak kabul edilebilir [65].

Örnekler arasındaki uyumu sayısal olarak ifade etmek için kullanılan kappa değerinin 0,52 olduğu ve bu değerin örneklerin ekseriyetle uyumlu olduğunu ve tahminin şans eseri yapılmadığını göstermektedir. Bunun yanında genellikle en az 0,5 olarak belirlenen kabul edilebilir F-ölçütü değerinin bu veri kümesinde 0,89 olmasının da ilgili modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

Veri kümesinde verilen örnekler bilgi sağlamak amacıyla kullanılmaktadır. Sadece bilginin yeterli miktarda örnek kullanılarak elde ediliyor olması, gerçek anlamda doğruluğunu sağlayabilir. Bilgi veri üzerinden alındığına göre örnek miktarındaki yetersizlik bilgi de de bir eksikliğe yol açacaktır. Kullanılan örnek miktarı yeterli değilse elde edilen bilginin eksikliğinden de söz edilebilir. Bunun için örneklerin ilgili model için

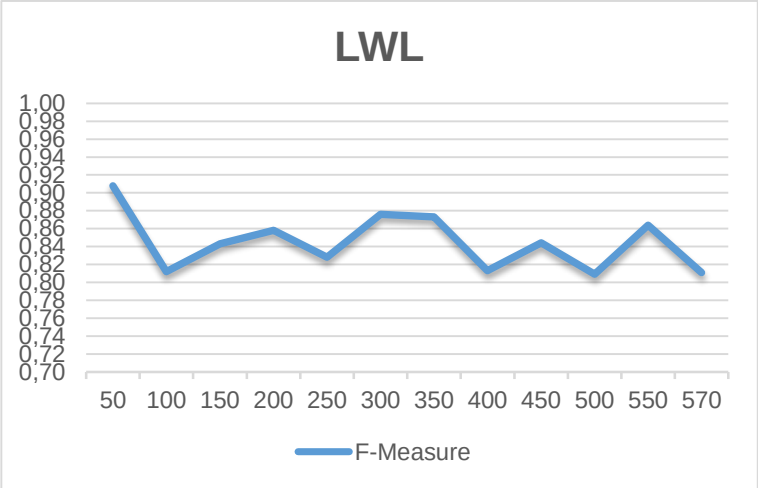
yeterli doygunlukta olduğunu gözlemlemek gerekir. Bu nedenle elde edilen sonuçlardan f-ölçütü – örnek sayısı değerleri kullanılarak çizilen grafik incelenmelidir.

Naïve Bayes sınıflandırma algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü – örnek sayısı değerlerinin işaretlenmesi ile oluşturulan grafik incelenmiştir. Grafik incelendiğinde f-ölçütü değerinin 50 örnekli veri kümesinden itibaren örnek sayısındaki artışla birlikte olumlu yönde arttığı ve 300 örneğe eriştiğinde aldığı f-ölçüt değerini 450 örneğe sahip veri kümesine kadar koruduğu gözlenmektedir. Buradan deneylerde kullanılan verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmış olduğu sonucu çıkarılmaktadır. Bunun yanında 400 örnekli alt veri kümesinde görülen dalgalanmanın ilgili alt veri kümesindeki kirlilikten kaynaklandığı söylenebilir.

4.4.2. LWL ile Doygunluk Deneyleri

Tablo 4.6 LWL Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
Lazy.LWL	50	0,90	0,61	0,93	0,90	0,91	0,25
Lazy.LWL	100	0,86	0,09	0,82	0,86	0,81	0,30
Lazy.LWL	150	0,89	0,00	0,80	0,89	0,84	0,27
Lazy.LWL	200	0,85	0,56	0,87	0,85	0,86	0,31
Lazy.LWL	250	0,88	0,06	0,89	0,88	0,83	0,29
Lazy.LWL	300	0,89	0,42	0,88	0,89	0,88	0,30
Lazy.LWL	350	0,89	0,39	0,87	0,89	0,87	0,28
Lazy.LWL	400	0,86	0,13	0,84	0,86	0,81	0,30
Lazy.LWL	450	0,89	0,07	0,90	0,89	0,84	0,28
Lazy.LWL	500	0,87	0,07	0,88	0,87	0,81	0,29
Lazy.LWL	550	0,89	0,30	0,87	0,89	0,86	0,28
Lazy.LWL	570	0,86	0,12	0,88	0,87	0,81	0,30



Şekil 4.9 LWL Algoritması Mutluluk Grafiği

Veri kümesinden, oransal olarak giderek büyüyen alt veri kümeleri seçilmiştir. 50 örnekten oluşan en küçük kümeden itibaren 50 şer arttırılarak 100, 150, 200, ….570 örnekten oluşan diğer veri kümeleri oluşturulmuştur. Bu veri kümeleri üzerinde LWL sınıflandırma algoritması kullanılarak elde edilen deney sonuçları incelendiğinde 50 örnekli alt veri kümesinin performansı en yüksek alt veri kümesi olduğu görülmektedir. Doğru sınıflandırma başarısı %90 değerindedir. En az 0,5 olarak belirlenen kabul edilebilir F-ölçütü değerinin 50 örnekli veri kümesinde 0,91 değerini almıştır. Bu veri kümesi için ilgili modelin sınıflandırma tanısı için yeterli olduğu rahatlıkla söylenebilir. Tüm alt kümeler için elde edilen sonuçlar bir grafik üzerinde toplanmıştır. Bu grafik incelendiğinde en yüksek doğru sınıflandırma başarısının en az örnekle yapılan öğrenme sonucunda çıkması ve daha fazla örnekle yapılan öğrenmelerin sınıflandırma başarısının inişli çıkışlı değerler alması nedeniyle sonucun rastlantısal olduğunu söylemek yanlış olmayacaktır.

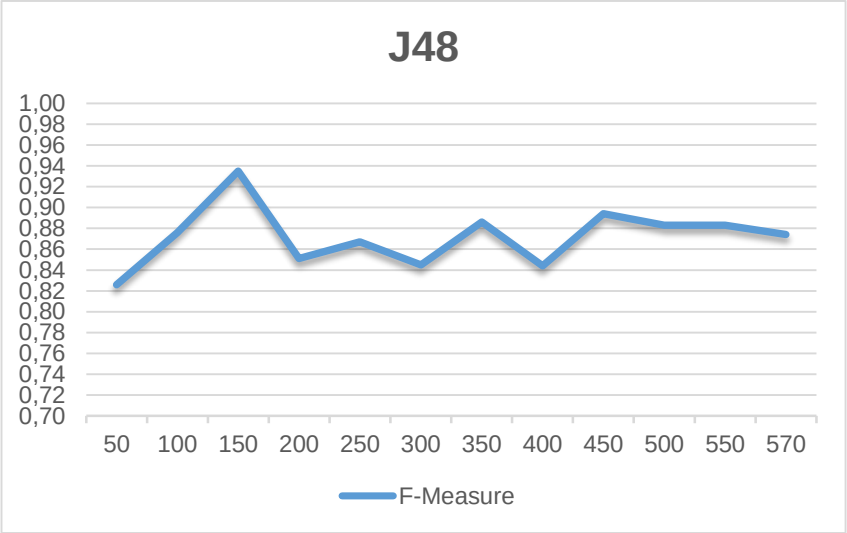
Örnekler arasındaki uyumu sayısal olarak ifade etmek için kullanılan kappa değerinin 0,61 olması ilgili alt veri kümesindeki örneklerin önemli derecede uyuşma içinde olduğunu ve tahminin şans eseri yapılmadığını göstermektedir. LWL algoritmasının alt kümelerin tümünde uygulanması ile elde edilen sonuçlar bir tabloda toplanmıştır. Bu tablo incelendiğinde yapılan on iki deneyden üçünün kappa değerinin 0.42 nin üstünde olduğu, geri kalan dokuz deney için kappa değerlerinin 0.39 un altında kaldığı görülmüştür. Deneylerin çoğunluğunda kappa değerinin belirlenen sınırın altında kalması sınıflandırmanın şans eseri olduğu sonucunu vermektedir.

LWL sınıflandırma algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü – örnek sayısı grafiği incelenmiştir. Buna göre, f-ölçütü değerinin 50 örnekli alt veri kümesinde en yüksek değerini aldığı görülmüştür. Örnek sayısının artmasıyla grafikte f-ölçütü sürekli bir dalgalanma göstermektedir. Bu dalgalanma deneylerde kullanılan verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmamış olmasıyla açıklanabilir. Teorik olarak, eğitim için kullanılan örneklerin artmasının sınıflandırma başarısını olumlu yönde etkilemesi beklenmektedir. Ancak mevcut veri kümesi için LWL algoritması örneklerin artması ile olumlu yönde etkilenmemiştir. Bu etki algoritmanın veri kümesindeki kirliliğe aşırı uyum göstermesi nedeniyle, algoritmanın veri kümesi ile uyumsuz olması nedeniyle veya verinin yetersiz olması nedeniyle oluşmuş olabilir.

4.4.3. J48 ile Doygunluk Deneyleri

Tablo 4.7 J48 Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
trees.J48	50	0,84	0,12	0,81	0,84	0,83	0,36
trees.J48	100	0,88	0,47	0,87	0,88	0,88	0,34
trees.J48	150	0,94	0,63	0,94	0,94	0,94	0,25
trees.J48	200	0,86	0,49	0,85	0,86	0,85	0,36
trees.J48	250	0,88	0,35	0,86	0,88	0,87	0,32
trees.J48	300	0,86	0,30	0,84	0,86	0,85	0,34
trees.J48	350	0,89	0,46	0,88	0,89	0,89	0,30
trees.J48	400	0,86	0,33	0,84	0,86	0,84	0,34
trees.J48	450	0,91	0,42	0,89	0,91	0,89	0,28
trees.J48	500	0,89	0,49	0,88	0,89	0,88	0,31
trees.J48	550	0,89	0,44	0,88	0,89	0,88	0,30
trees.J48	570	0,88	0,46	0,87	0,88	0,87	0,31



Şekil 4.10 J48 Algoritması Mutluluk Grafiği

En küçük veri kümesinin 50 örnekten oluştuğu daha sonra 100 örneğe sahip veri seti ve 50 örnek arttırarak oluşturulan 150, 200, ….550 olarak sıralanan ve en son 570 örnekten oluşan alt veri kümeleri oluşturulmuştur. Bu veri kümeleri üzerinde J48 algoritması kullanılarak elde edilen deney sonuçları incelenmiştir. En yüksek örneğe sahip 570 örnekli alt veri kümesi %88 doğru sınıflandırma oranına sahipken, 450 örnekli alt veri kümesi %91, 150 örnekli alt veri kümesi ise %94 doğru sınıflandırma oranına sahiptir. En yüksek performansı 150 örneğe sahip olan alt veri kümesi göstermiştir. Bu alt veri kümesi için F-ölçütü değerinin 0,94 olması J48 algoritmasının oluşturduğu bu modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir. 150 örnekle yapılan öğrenmeden elde edilen sınıflandırma işleminin daha fazla örnekle yapılan öğrenme sonucunda yapılan sınıflandırma işlemlerinden daha başarılı olması nedeniyle elde edilen başarı sonucunun şans eseri olduğu söylenebilir.

Örnekler arasındaki uyumu sayısal olarak ifade etmek için kullanılan kappa değerinin 150 örnekli alt veri kümesi için 0,63 olduğu görülmüştür. Bu değer 150 örnekli alt veri kümesindeki örneklerin arasında önemli derecede uyuşma olduğunu ve bu nedenle tahminin şans eseri yapılmadığını göstermektedir.

J48 algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü – örnek sayısı ilişkisi grafiği verilmiştir. 570 örnekli alt veri kümesinin f-ölçütü değerinin 0,87 olduğu, 450 ve 350 örnekli alt veri kümelerinin f-ölçütü değerlerinin 0,89 olduğu, 150 örnekli alt veri kümesinin f-ölçütü değerinin ise 0,94 olduğu grafikten görülmektedir. En yüksek f-ölçütü değerini 150 örnekli alt veri kümesinin aldığı görülmektedir. Örnek

sayısının arttırılmasıyla f-ölçütü değerinin artması beklenirken grafik sürekli bir dalgalanma göstermiştir. Bu deneylerde kullanılan verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığı şeklinde yorumlanmaktadır. Bu durumun sebebi J48 algoritmasının veri kümesindeki kirliliğe aşırı uyum göstermesi ile veya algoritmanın ilgili veri kümeleri için uyumsuz olması şeklinde açıklanabilir.

4.4.4. Jrip ile Doygunluk Deneyleri

Tablo 4.8 Jrip Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
Rules.Jrip	50	0,92	0,46	0,95	0,92	0,93	0,25
Rules.Jrip	100	0,86	0,38	0,85	0,86	0,86	0,34
Rules.Jrip	150	0,89	0,41	0,89	0,89	0,89	0,31
Rules.Jrip	200	0,86	0,54	0,86	0,86	0,86	0,32
Rules.Jrip	250	0,90	0,57	0,91	0,90	0,91	0,28
Rules.Jrip	300	0,87	0,41	0,86	0,87	0,87	0,32
Rules.Jrip	350	0,90	0,53	0,90	0,90	0,90	0,29
Rules.Jrip	400	0,87	0,45	0,86	0,87	0,87	0,33
Rules.Jrip	450	0,93	0,61	0,93	0,93	0,93	0,25
Rules.Jrip	500	0,90	0,56	0,89	0,90	0,90	0,29
Rules.Jrip	550	0,91	0,56	0,90	0,91	0,91	0,28
Rules.Jrip	570	0,89	0,52	0,88	0,89	0,88	0,31



Şekil 4.11 Jrip Algoritması Mutluluk Grafiği

Jrip algoritması ile deneyi yapılacak veri kümelerinin ilki 50 örneğe sahip olacak şekilde seçilmiştir. Daha sonra 50 şer arttırılarak 100 örnekli, 150 örnekli ….550 örnekli olarak giden ve en son 570 örnekten oluşan alt veri kümeleri oluşturulmuştur. Bu veri kümeleri üzerinde Jrip algoritması kullanarak elde edilen deney sonuçları doğru sınıflandırma oranı, precision, recall, f-ölçütü, RMSE değerlerinin yer aldığı bir tablo oluşturulmuştur. Deneyler sonucunda tabloda yer alan değerlerden doğru sınıflandırma oranı dikkate alınarak yapılan incelemeye göre 550 örnekli alt veri kümesi %91 doğru sınıflandırma oranına sahipken, 50 örnekli alt veri kümesi %92 ve 450 örnekli alt veri kümesi %93 doğru sınıflandırma oranına sahiptir. Performansı en yüksek alt veri kümesinin %93 doğru sınıflandırma oranı ile 450 örneğe sahip olan veri kümesi olduğu görülmektedir. JRIP algoritması kullanılarak oluşturulan bu sınıflandırma modelinin 450 örnekli alt veri kümesi için 0,93 F-ölçütü değerine sahip olması modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

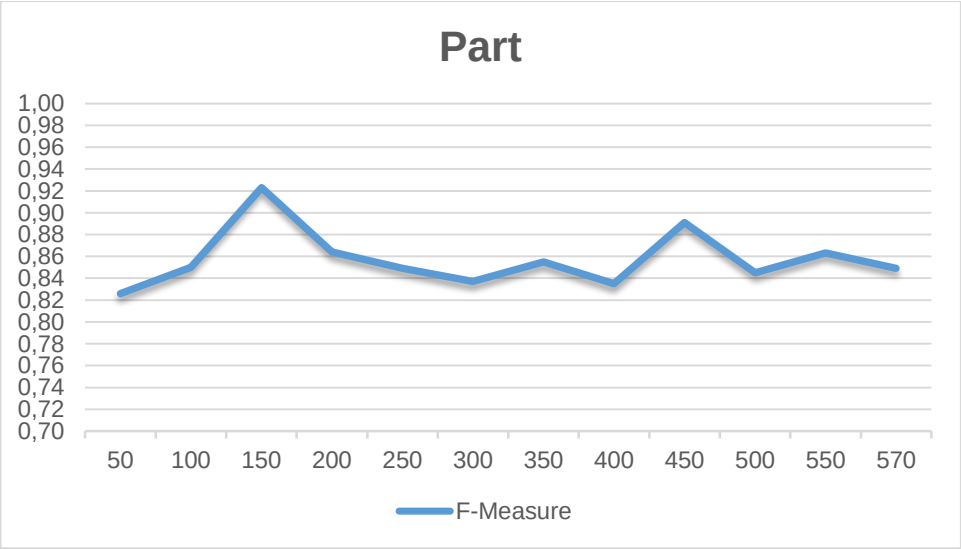
Deneyler sonucunda oluşturulan tablodan kappa değerleri dikkate alınarak inceleme yapıldığında en yüksek doğru sınıflandırma performansına sahip 450 örnekli alt veri kümesinin kappa değerinin 0.61 olduğu görülmüştür. Kappa değeri örnekler arasındaki uyumu gösterdiğinden, 450 örnekli alt veri kümesinin JRIP algoritması ile yapılan sınıflandırma sonucunun şans eseri olmadığı göstermektedir. Kappa değerinden yola çıkılarak bu veri kümesi için örneklerin birbiriyle oldukça uyumlu olduğu söylenebilir.

Jrip algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü – örnek sayısı ilişkisinin gösterildiği mutluluk grafiği incelendiğinde ise f-ölçütü değerinin 450 örnekli alt veri kümesinde en yüksek değerini aldığı görülmektedir. Mutluluk grafiğinin tamamı incelendiğinde örnek sayısındaki artış ile f-ölçütü değerlerinin artması beklenirken bu değerde sürekli bir dalgalanma olduğu görülmektedir. Bu dalgalanmanın sebebi veri yetersizliği olabileceği gibi performans sonuçlarının şans eseri alınmış olması da olabilir. Deneylerde kullanılan alt veri kümelerinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığını sonucu bu grafikten çıkarılabilir. Jrip algoritmasının veri kümesindeki kirliliğe aşırı uyum göstermesi veya algoritmanın ilgili alt veri kümeleri için uygun olmadığı sonucuna ulaşılabilir.

4.4.5. Part Algoritması ile Doygunluk Deneyleri

Tablo 4.9 Part Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
Rules.Part	50	0,84	0,12	0,81	0,84	0,83	0,36
Rules.Part	100	0,86	0,34	0,84	0,86	0,85	0,34
Rules.Part	150	0,93	0,58	0,92	0,93	0,92	0,27
Rules.Part	200	0,87	0,53	0,86	0,87	0,86	0,34
Rules.Part	250	0,85	0,29	0,85	0,85	0,85	0,36
Rules.Part	300	0,85	0,28	0,83	0,85	0,84	0,37
Rules.Part	350	0,86	0,32	0,85	0,86	0,86	0,35
Rules.Part	400	0,84	0,32	0,83	0,84	0,84	0,37
Rules.Part	450	0,90	0,42	0,89	0,90	0,89	0,30
Rules.Part	500	0,84	0,37	0,85	0,84	0,85	0,36
Rules.Part	550	0,87	0,35	0,86	0,87	0,86	0,34
Rules.Part	570	0,86	0,36	0,84	0,86	0,85	0,35



Şekil 4.12 Part Algoritması Mutluluk Grafiği

570 örnekten oluşan veri kümesi en küçüğü 50 örneğe sahip alt veri kümesi olmak üzere 50 şer artarak 100,150,,…550 ve son olarak 570 örnekli alt veri kümesi olacak şekilde oluşturulmuştur. Oluşturulan alt veri kümelerinin her biri için Part algoritması

kullanılarak elde edilen sonuçlar bir tabloda toplanmıştır. Tablonun doğru sınıflandırma oranlarını içeren sütunu incelendiğinde göze çarpan doğru sınıflandırma oranları 450 örnekli alt veri kümesi ile 150 örnekli alt veri kümesinin sahip olduğu oranlardır. Bunlardan 150 örnekli alt veri kümesinin doğru sınıflandırma oranı %93 olarak görülmektedir. En yüksek doğru sınıflandırma oranına sahip 150 örnekli alt veri kümesinin F-ölçütü değerinin 0,92 olması sayesinde Part algoritmasının bu veri kümesi ile yaptığı sınıflandırma modelinin sınıflandırma tanısı için yeterli olduğu anlaşılmaktadır. Algoritmanın tahmin gücünü sayısal olarak ifade etmek için kullanılan kappa değerinin en yüksek performansa sahip olan 150 örnekli alt veri kümesi için 0,58 olduğu görülmektedir. Bu değer ilgili alt veri kümesindeki örneklerin ekseriyetle uyuşma içinde olduğunu ve Part algoritması ile yapılan tahminin şans eseri olmadığını göstermektedir. Ancak daha yüksek örnekle yapılan öğrenme sonuçlarının 150 örnekli alt veri kümesi ile yapılan öğrenme sonucundan daha düşük olması yapılan sınıflandırma işlemlerinde elde edilen başarının tesadüfi olduğu sonucunu doğurmaktadır.

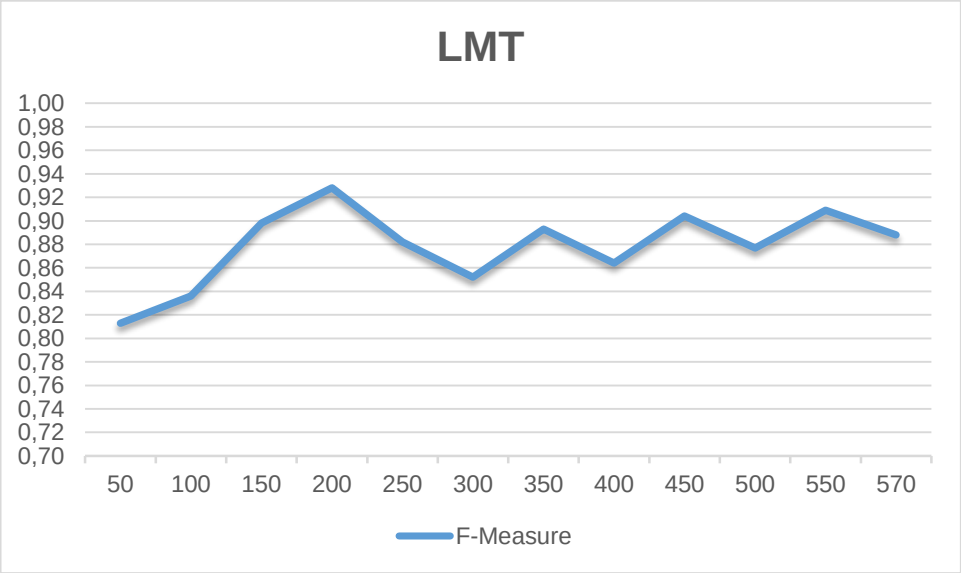
Part algoritmasının alt veri kümelerinde kullanılması ile elde edilen mutluluk grafiği incelendiğinde, f-ölçütü değerinin 0,83 ile 0,92 arasında değiştiği görülmektedir. En yüksek f-ölçütü değerinin en yüksek doğruluk oranına sahip 150 örnekli alt veri kümesinde olduğu görülmektedir. Örnek sayısının artmasıyla grafikte f-ölçütü değerinin 50,100 ve 150 örnekli alt veri kümeleri için bir yükselme içinde olduğu, 200 örnekte keskin bir düşüşten sonra sürekli bir dalgalanma halinde olduğu görülmektedir. Bu dalgalanma deneylerde kullanılan verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığını göstermektedir. Part algoritmasının gösterdiği dalgalanmalı durum için Part algoritmasının veri kümesindeki kirliliğe aşırı uyum gösterdiği ve algoritmanın veri kümesi için uyumsuz olduğu söylenebilir.

4.4.6. LMT Algoritması ile Doygunluk Deneyleri

Tablo 4.10 LMT Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
trees.LMT	50	0,82	0,08	0,81	0,82	0,81	0,35
trees.LMT	100	0,85	0,27	0,83	0,85	0,84	0,35
trees.LMT	150	0,91	0,40	0,90	0,91	0,90	0,33
trees.LMT	200	0,93	0,75	0,93	0,93	0,93	0,26

trees.LMT	250	0,90	0,41	0,88	0,90	0,88	0,28
trees.LMT	300	0,86	0,33	0,85	0,86	0,85	0,34
trees.LMT	350	0,90	0,49	0,89	0,90	0,89	0,28
trees.LMT	400	0,87	0,42	0,86	0,87	0,86	0,31
trees.LMT	450	0,91	0,49	0,90	0,91	0,90	0,26
trees.LMT	500	0,89	0,46	0,87	0,89	0,88	0,29
trees.LMT	550	0,91	0,56	0,91	0,92	0,91	0,27
trees.LMT	570	0,89	0,52	0,89	0,90	0,89	0,28



Şekil 4.13 LMT Algoritması Mutluluk Grafiği

Veri kümesi kullanılarak en küçüğü 50 örneğe sahip alt veri kümesi olmak üzere 50 şer artarak 100,150,,…550 ve son olarak 570 örnekli oniki adet alt veri kümesi oluşturulmuştur. Bu veri kümeleri üzerinde LMT algoritması kullanılarak elde edilen deney sonuçları bir tabloda toplanmıştır. Doğru sınıflandırma oranına göre alt veri kümeleri sıralandığında en düşük değer %82 ile 100 örnekli alt veri kümesine aitken, performansı en yüksek alt veri kümesinin %93 doğru sınıflandırma oranı ile 200 örneğe sahip olan veri kümesi olduğu görülmektedir. Tablo f-ölçütü değerleri için incelendiğinde bu değerlerin 0,81 ile 0,93 arasında değiştiği görülmektedir. F-ölçütü değeri 200 örnekli veri kümesi için en yüksek değeri almıştır. Bu LMT algoritmasının oluşturduğu bu modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

Algoritmanın tahmin gücünü sayısal olarak ifade etmek için kullanılan kappa değerinin 12 örnekten 3'ü için 0,33 değerinin altında, kalan 9 örnek için 0,4 değerinin

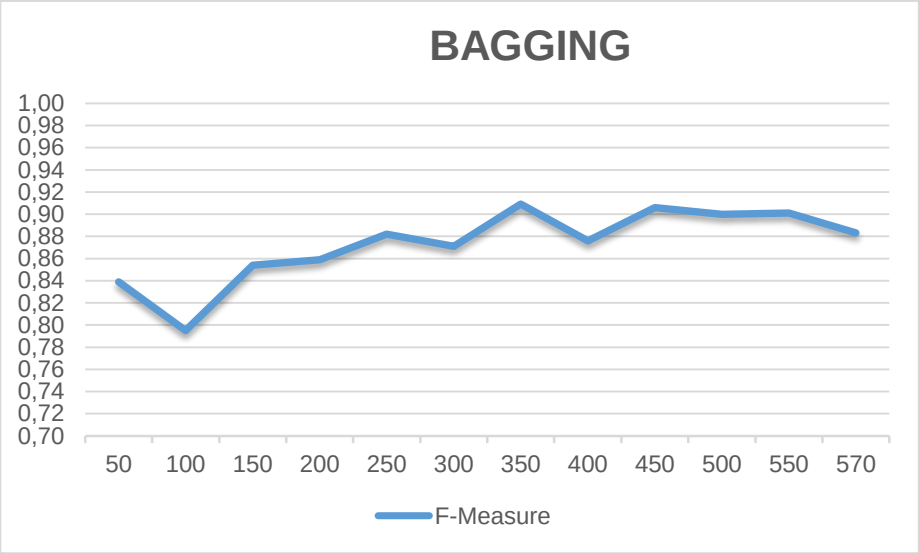
üstünde olduğu görülür. 200 örnekli alt veri kümesinin kappa değerinin 0,75 olması bu kümedeki örnekler arasında önemli derecede uyuşma olduğunu ve yapılan tahminin tesadüfi yapılmadığını göstermektedir. Ancak yine de en yüksek değerin az sayıda örnekte elde edilmesinden dolayı yapılan sınıflandırma işlemlerinde elde edilen başarının şans eseri olduğunu söylemek mümkündür.

LMT algoritmasının alt veri kümelerinde kullanılması ile elde edilen değerler mutluluk grafiği adı verilen örnek sayısı ile f-ölçütü ilişkisini gösteren bir grafiğe dökülmüştür. Grafikten f-ölçütü değerinin 200 örnekli alt veri kümesinde en yüksek değeri aldığı görülmektedir. Grafikte f-ölçütü değeri 50,100, 150 ve 200 örnekli alt veri kümeleri için yükselme eğiliminde iken 200 örnekli alt veri kümesinden sonra keskin bir şekilde düşmüş ve sonrasında sürekli bir dalgalanma göstermiştir. Veri örnekleme yoğunluğunun doygunluk seviyesine ulaşamadığı açıkça görülmektedir. Bunun sebebi algoritmanın veri kümesindeki kirliliğe aşırı uyum göstermesi olabileceği gibi, bu algoritmanın bu veri kümeleri için uygun olmaması da olabilir.

4.4.7. Bagging Algoritması ile Doygunluk Deneyleri

Tablo 4.11 Bagging Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
meta.Bagging	50	0,86	0,15	0,83	0,86	0,84	0,27
meta.Bagging	100	0,86	0,00	0,74	0,86	0,80	0,32
meta.Bagging	150	0,89	0,09	0,86	0,89	0,85	0,27
meta.Bagging	200	0,87	0,52	0,86	0,87	0,86	0,32
meta.Bagging	250	0,90	0,41	0,88	0,90	0,88	0,28
meta.Bagging	300	0,89	0,39	0,88	0,89	0,87	0,30
meta.Bagging	350	0,92	0,56	0,91	0,92	0,91	0,26
meta.Bagging	400	0,89	0,47	0,87	0,89	0,88	0,30
meta.Bagging	450	0,92	0,49	0,91	0,92	0,91	0,26
meta.Bagging	500	0,91	0,57	0,90	0,91	0,90	0,28
meta.Bagging	550	0,91	0,52	0,90	0,91	0,90	0,27
meta.Bagging	570	0,89	0,51	0,88	0,89	0,88	0,29



Şekil 4.14 Bagging Algoritması Mutluluk Grafiği

En küçüğü 50 örneğe sahip alt veri kümesi olmak üzere 50 şer artarak 100,150,,…550 ve son olarak 570 örnekli toplam oniki adet alt veri kümesi oluşturulmuştur. Bagging algoritması 12 alt veri kümesi için çalıştırılmış ve elde edilen veriler bir tabloda toplanmıştır. Bu tablodan yararlanılarak 12 alt veri kümesi için doğru sınıflandırma oranı değerlerinin %87 ile %92 arasında değiştiği söylenebilir. Deney sonuçları arasında performansı en yüksek alt veri kümelerinin %92 doğru sınıflandırma oranı ile 350 ve 450 örneğe sahip olan alt veri kümeleri olduğu görülmektedir. İki algoritmanın doğru sınıflandırma oranlarının eşit çıkmasından dolayı en iyi performansı hangisinin elde ettiğini belirlemek için örnek uyumuna bakılmasında fayda olacaktır. Bagging algoritmasının bu iki algoritmada çalışması sonucu aldığı değerler incelendiğinde 350 örneğe sahip alt veri kümesinin sahip olduğu kappa değerinin 0,56 olduğu 450 örneğe sahip veri kümesinin aldığı kappa değerinin 0,49 olduğu görülmektedir. Bu durum 350 örnekli alt veri kümesinin tahmin gücünün, 450 örnekli alt veri kümesinin tahmin gücüne göre daha yüksek olduğunu göstermektedir. Sonuç olarak 350 örneğe sahip olan alt veri kümesinin en başarılı performansı gösterdiği söylenebilir. Algoritmanın tahmin gücünü sayısal olarak ifade etmek için kullanılan kappa değerinin 0,56 olması bu değerin ilgili alt veri kümesindeki örneklerin ekseriyetle uyuştuğunu göstermekte ve tahminin şans eseri yapılmadığını sonucuna götürmektedir. Modelin sınıflandırma tanısı için yeterli olduğunun göstergesi olan F-ölçütü değerinin bu alt veri

kümesinde 0,91 olması ilgili modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

Bagging algoritmasının 50 örneğe sahip alt veri kümesinden başlanarak, 570 örneğe sahip alt veri kümesine kadar yapılan deneylerin sonucunda elde edilen f-ölçütü değerleri ile örnek sayısı ilişkilendirilerek oluşturulan mutluluk grafiği verilmiştir. Burada f-ölçütünün değeri, örnek sayısının 50 olduğu ilk alt veri kümesinde tablo geneline göre yüksek bir değer bulunurken 100 örneğe sahip alt veri kümesine gelindiğinde ani bir düşüş görülmektedir. 150 örnekli alt veri kümesinde tekrar ani yükselme görülmüştür. Bu noktadan sonra eğri genellikle artan veri sayısı ile birlikte yükselmeye eğilimi göstermektedir. 300 ile 400 örnekli veri setleri arasında yine ani bir yükselme ve ardından azalma görülmektedir. Elde edilen başarının şans eseri olup olmadığını anlamak için kappa değerlerine bakmak faydalı olacaktır. 50 örnekli alt veri kümesi için kappa değeri 0,15 iken, 100 örnekli alt veri kümesi için ise kappa değeri 0 ve 150 örnekli alt veri kümesi için ise kappa değeri 0,09 olmaktadır. Başarının tesadüfi olduğu bu değerlerden anlaşılmaktadır. Ayrıca veri sayısının yetersiz olduğu çıkarımı da yapılabilir.

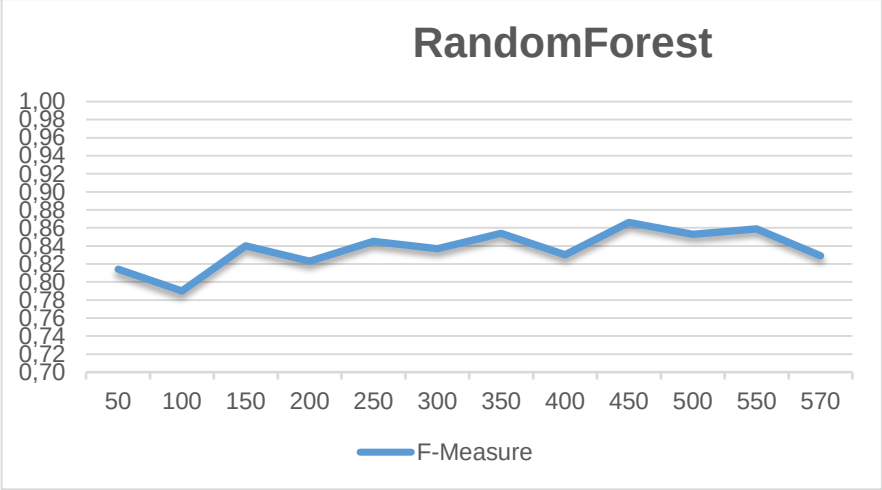
Bagging algoritması ile elde edilen mutluluk grafiği incelendiğinde, grafiğin tümünde dalgalanma olduğu ve f-ölçütü değerinin en yüksek olduğu alt veri kümesinin 350 örnekli alt veri kümesi olduğu görülmektedir. Bu durum deneylerde kullanılan verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığını göstermektedir. Bagging algoritmasının veri kümesindeki kirliliğe aşırı uyum göstermesi ihtimali ve algoritmanın ilgili alt veri kümeleri için uyumsuz olması bu durumu açıklamaktadır.

4.4.8. RandomForest Algoritması ile Doygunluk Deneyleri

Tablo 4.12 RandomForest Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
trees.RandomForest	50	0,86	-0,04	0,77	0,86	0,81	0,31
trees.RandomForest	100	0,85	-0,02	0,74	0,85	0,79	0,31
trees.RandomForest	150	0,89	-0,01	0,80	0,89	0,84	0,29
trees.RandomForest	200	0,86	0,35	0,85	0,86	0,82	0,31
trees.RandomForest	250	0,89	0,16	0,90	0,89	0,85	0,29
trees.RandomForest	300	0,88	0,19	0,90	0,88	0,84	0,30
trees.RandomForest	350	0,89	0,25	0,87	0,89	0,85	0,29

trees.RandomForest	400	0,87	0,21	0,85	0,87	0,83	0,30
trees.RandomForest	450	0,90	0,21	0,90	0,90	0,87	0,27
trees.RandomForest	500	0,88	0,30	0,88	0,88	0,85	0,29
trees.RandomForest	550	0,89	0,26	0,90	0,89	0,86	0,28
trees.RandomForest	570	0,87	0,21	0,86	0,87	0,83	0,30



Şekil 4.15 Bagging Algoritması Mutluluk Grafiği

Random Forest algoritması ile deneyi yapılacak veri kümelerinin ilki 50 örneğe sahip olacak şekilde seçilmiştir. Daha sonra 50 şer arttırılarak 100 örnekli, 150 örnekli ….550 örnekli olarak giden ve en son 570 örnekten oluşan alt veri kümeleri oluşturuldu. Bu veri kümeleri üzerinde Random Forest algoritması kullanarak elde edilen deney sonuçları bir tabloya aktarılmıştır. Deneyler sonucunda doğru sınıflandırma oranına göre bir sıralama yapıldığında 100 ve 50 örnekli alt veri kümeleri sırasıyla %85 ve %86 doğruluk oranlarıyla en alt sırada yer alırken 150, 250, 350 ve 550 örnekli alt veri kümeleri ise %89 doğruluk oranıyla üst sıralarda yer almaktadır. Performansı en yüksek alt veri kümesinin %90 doğru sınıflandırma oranı ile 450 örneğe sahip olan alt veri kümesi olduğu görülmektedir. F-ölçütü değerinin 450 örnekli alt veri kümesinde 0,87 değerini alması RandomForest algoritmasının oluşturduğu bu modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

Örnekler arasındaki uyumu sayısal olarak ifade etmek ve tahminin şans eseri olup olmadığının sınanması için kullanılan kappa değerinin en iyi performansı gösteren 450 örnekli alt veri kümesi için 0,21 değerini aldığı görülmektedir. Bu değerin düşük olması bu alt veri kümesinde bulunan örneklerin arasındaki uyuşmanın önemsiz olduğunu ve

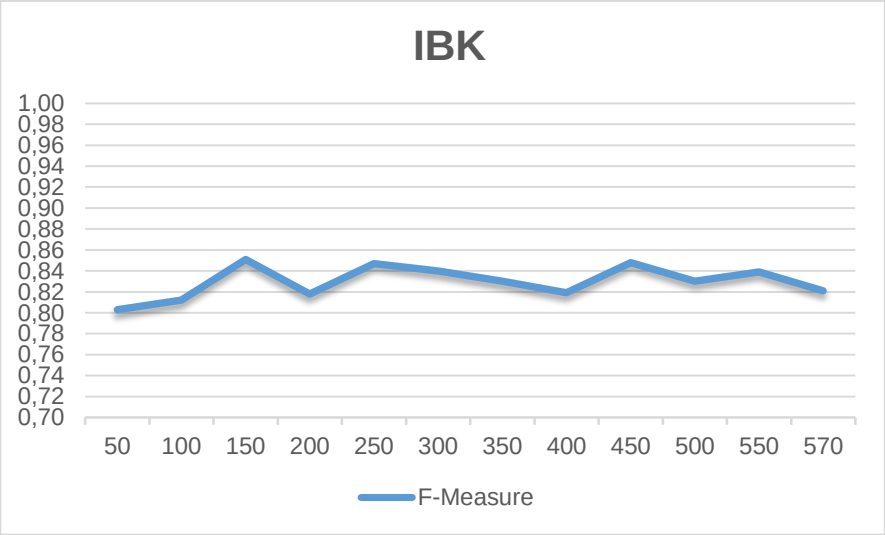
tahminin şans eseri yapıldığını göstermektedir. Tablonun tamamına bakıldığında en yüksek kappa değerinin 200 örneğe sahip alt veri kümesi tarafından 0,35 olarak alındığı görülmektedir. Bu da ilgili alt veri kümesi için orta derecede uyuşma olduğunu göstermektedir.

Random Forest algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü örnek sayısı ilişkisinin gösterildiği mutluluk grafiği verilmiştir. Bu grafik incelendiğinde f-ölçütü değerinin 450 örnekli alt veri kümesinde en yüksek değerini aldığı görülmektedir. Mutluluk grafiğinin tamamı incelendiğinde örnek sayısının her artışında f-ölçütü değerinde aşırı miktarda dalgalanma olduğu görülmektedir. Eldeki veriler ışığında verinin yeterli olmadığı ve performans sonuçlarının şans eseri olduğu rahatlıkla söylenebilir. Deneylerde kullanılan alt veri kümelerinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığı sonucunu çıkarılabilir. Algoritmanın bu sonuçları vermesinin nedeni olarak algoritmanın veri kümesindeki kirliliğe aşırı uyum göstermesi, algoritmanın ilgili alt veri kümeleri için uyumsuz olması söylenebilir.

4.4.9. IBK Algoritması ile Doygunluk Deneyleri

Tablo 4.13 IBK Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
Lazy.IBK	50	0,84	-0,06	0,77	0,84	0,80	0,37
Lazy.IBK	100	0,84	0,13	0,80	0,84	0,81	0,39
Lazy.IBK	150	0,87	0,12	0,84	0,87	0,85	0,35
Lazy.IBK	200	0,84	0,35	0,81	0,84	0,82	0,39
Lazy.IBK	250	0,88	0,20	0,84	0,88	0,85	0,34
Lazy.IBK	300	0,86	0,24	0,83	0,86	0,84	0,37
Lazy.IBK	350	0,86	0,14	0,82	0,86	0,83	0,36
Lazy.IBK	400	0,84	0,20	0,81	0,84	0,82	0,39
Lazy.IBK	450	0,88	0,13	0,84	0,88	0,85	0,34
Lazy.IBK	500	0,86	0,21	0,82	0,86	0,83	0,37
Lazy.IBK	550	0,87	0,19	0,83	0,87	0,84	0,35
Lazy.IBK	570	0,85	0,20	0,81	0,85	0,82	0,38



Şekil 4.16 IBK Algoritması Mutluluk Grafiği

Veri kümesinin sahip olduğu örnekler, ilki 50 örneğe sahip olacak şekilde sonrakileri 50 şer arttırılarak 100 örnekli, 150 örnekli ….550 örnekli olarak artan ve en son 570 örnekten oluşan alt veri kümeleri oluşturacak şekilde seçilmiştir. Bu alt veri kümeleri üzerinde IBK algoritması kullanarak elde edilen deney sonuçları performans tablosuna girilmiştir.

Deneyler sonucunda performansı en yüksek alt veri kümesinin %88 doğru sınıflandırma oranı ile 250 ve 450 örneğe sahip olan veri kümesi olduğu görülmektedir. Örnekler arasındaki uyumu sayısal olarak ifade etmek ve tahminin şans eseri olup olmadığının sınanması için kullanılan kappa değerinin 250 örneğe sahip alt veri kümesinde, 450 örneğe sahip alt veri kümesinden daha yüksektir. Bu IBK algoritmasının bu modelinin tahmin gücünün 250 örnekli alt veri kümesinde daha yüksek olduğunun göstergesidir. Kappa değerinin 0,4 ün altında olması örnek uyuşmazlığını gösterdiği için 250 örnekli alt veri kümesinin kappa değerinin 0,20 olması örnekler arasında önemsiz uyuşma olduğu şeklinde ifade edilir. Aynı zamanda performanstaki bu başarı oranlarının şans eseri alındığını göstermektedir.

IBK algoritmasının alt veri kümelerinde kullanılması ile elde edilen sonuçların tümüne bakıldığında eldeki kappa değerleri için IBK algoritması için veri kümesinin tamamının dengesiz olduğundan bahsedilebilir. Bu nedenle performans sonuçlarının şans eseri olduğu kabul edilebilir.

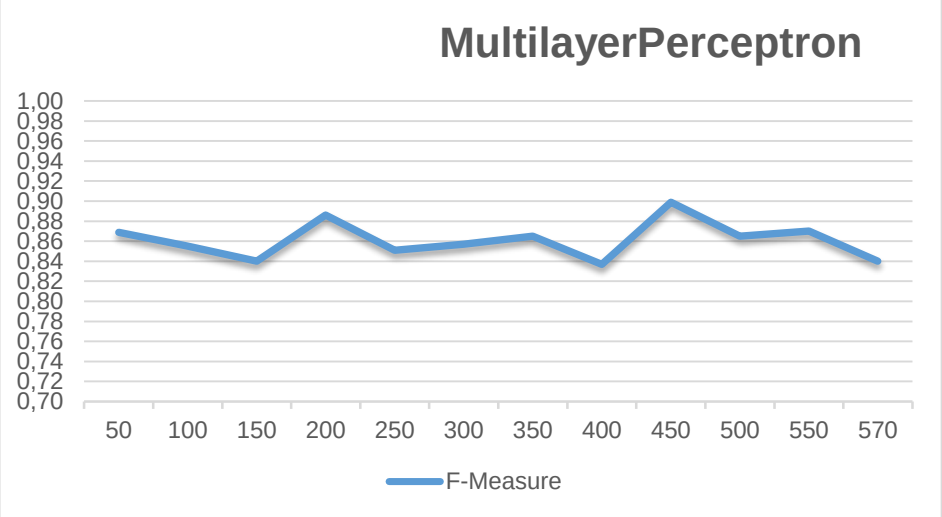
IBK algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü örnek sayısı ilişkisini gösteren mutluluk grafiği incelendiğinde ise 150, 250 ve 450 örnekli alt

veri kümelerinde f-ölçütü değerinin en yüksek değerini aldığı görülmektedir. Mutluluk grafiğinin tamamı incelendiğinde deneylerde kullanılan alt veri kümelerinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığı söylenebilir. Algoritmanın bu sonuçları vermesinin nedeni olarak veri kümesindeki dengesizlik, algoritmanın kirliliğe aşırı uyum göstermesi, algoritmanın ilgili alt veri kümeleri için uyumsuz olabileceği de söylenebilir.

4.4.10. MultilayerPerceptron Algoritması ile Doygunluk Deneyleri

Tablo 4.14 MultilayerPerceptron Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
MultilayerPerceptron	50	0,88	0,34	0,86	0,88	0,87	0,32
MultilayerPerceptron	100	0,86	0,38	0,85	0,86	0,86	0,35
MultilayerPerceptron	150	0,85	0,12	0,83	0,85	0,84	0,36
MultilayerPerceptron	200	0,89	0,62	0,89	0,89	0,89	0,33
MultilayerPerceptron	250	0,85	0,31	0,85	0,85	0,85	0,37
MultilayerPerceptron	300	0,87	0,35	0,85	0,87	0,86	0,35
MultilayerPerceptron	350	0,87	0,37	0,86	0,87	0,87	0,35
MultilayerPerceptron	400	0,84	0,34	0,84	0,84	0,84	0,38
MultilayerPerceptron	450	0,90	0,48	0,90	0,90	0,90	0,30
MultilayerPerceptron	500	0,87	0,42	0,86	0,87	0,87	0,35
MultilayerPerceptron	550	0,87	0,39	0,87	0,88	0,87	0,34
MultilayerPerceptron	570	0,84	0,34	0,84	0,84	0,84	0,38



Şekil 4.17 MultilayerPerceptron Algoritması Mutluluk Grafiği

Multilayer Perceptron algoritması ile deneyi yapılacak veri kümelerinin ilki 50 örneğe sahip olacak şekilde seçilmiştir. Daha sonra 50 şer arttırılarak 100 örnekli, 150 örnekli ….550 örnekli olarak giden ve en son 570 örnekten oluşan alt veri kümeleri oluşturuldu. Bu veri kümeleri üzerinde Multilayer Perceptron algoritması kullanarak elde edilen deney sonuçları bir tabloya yazılmıştır. Doğru sınıflandırma performansına göre veri kümeleri incelendiğinde en kötü performansın %84 ile 400 ve 570 örnekli alt veri kümelerinde elde edildiği görülmektedir. En fazla örneğe sahip alt veri kümesinin en düşük performansa sahip olmasının veri kirliliğinden kaynaklandığı düşünülebilir. Performansı en yüksek alt veri kümesinin %90 doğru sınıflandırma oranı ile 450 örneğe sahip olan veri kümesi olduğu görülmektedir. Doğru sınıflandırma oranında en iyi performansa sahip 450 örnekli veri kümesinin 0,90 olan F-ölçütü değeri Multilayer Perceptron algoritmasının bu veri kümesi için sınıflandırma tanısı için yeterli olduğunu göstermektedir.

Algoritmanın tahmin gücünü sayısal olarak ifade etmek için kullanılan kappa değerinin 0,48 olması bu değerin ilgili alt veri kümesindeki örneklerin ekseriyetle uyuşma içinde olduğunu ve tahminin şans eseri yapılmadığını göstermektedir.

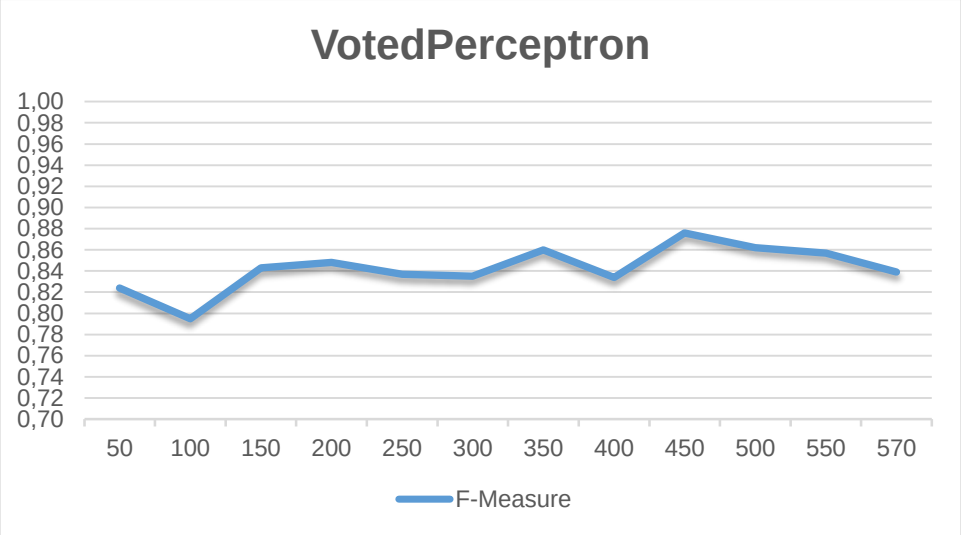
Multilayer Perceptron algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü örnek sayısı ilişkisinin gösterildiği mutluluk grafiği incelendiğinde ise f-ölçütü değerinin 450 örnekli alt veri kümesinde en yüksek değerini aldığı görülmektedir. Mutluluk grafiğinin tamamı incelendiğinde örnek sayısının artmasıyla f-ölçütü değerinde yüksek inişler ve çıkışlar dikkat çekmektedir. Bu iniş ve çıkışlar deneylerde kullanılan alt veri kümelerinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığını göstermektedir. Algoritmanın bu sonuçları vermesinin nedeni olarak algoritmanın veri kümesindeki kirliliğe aşırı uyum göstermesi, algoritmanın ilgili alt veri kümeleri için uyumsuz olması gösterilebilir.

4.4.11. VotedPerceptron Algoritması ile Doygunluk Deneyleri

Tablo 4.15 VotedPerceptrom Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
VotedPerceptron	50	0,88	0,00	0,77	0,88	0,82	0,35
VotedPerceptron	100	0,86	0,00	0,74	0,86	0,80	0,37
VotedPerceptron	150	0,89	0,00	0,80	0,89	0,84	0,33

VotedPerceptron	200	0,87	0,46	0,85	0,87	0,85	0,37
VotedPerceptron	250	0,88	0,13	0,84	0,88	0,84	0,35
VotedPerceptron	300	0,88	0,18	0,87	0,88	0,84	0,35
VotedPerceptron	350	0,89	0,29	0,87	0,89	0,86	0,34
VotedPerceptron	400	0,86	0,25	0,83	0,86	0,83	0,37
VotedPerceptron	450	0,90	0,29	0,88	0,90	0,88	0,31
VotedPerceptron	500	0,88	0,36	0,87	0,88	0,86	0,34
VotedPerceptron	550	0,88	0,28	0,85	0,88	0,86	0,35
VotedPerceptron	570	0,86	0,29	0,83	0,86	0,84	0,37



Şekil 4.18 VotedPerceptron Algoritması Mutluluk Grafiği

Veri kümesinin sahip olduğu örnekler, ilki 50 örneğe sahip olacak şekilde sonrakileri 50 şer arttırılarak 100 örnekli, 150 örnekli ….550 örnekli olarak artan ve en son 570 örnekten oluşan alt veri kümeleri oluşturacak şekilde seçilmiştir. Bu alt veri kümeleri üzerinde VotedPerceptron algoritması kullanarak elde edilen deney sonuçları performans tablosuna girilmiştir.

12 alt veri kümesinde VotedPerceptron algoritmasının çalıştırılmasıyla elde edilen doğru sınıflandırma oranları %86 ile %90 arasında değişmektedir. Tablodan performansı en yüksek alt veri kümesinin %90 doğru sınıflandırma oranı ile 450 örneğe sahip olan alt veri kümesi olduğu görülmektedir. Örnekler arasındaki uyumu sayısal olarak ifade etmek ve tahmin sonuçlarının şans eseri olup olmadığının sınanması için kullanılan kappa değerinin incelenmesi sonucu tablodajki değerlerin 0 ile 0,46 arasında değiştiği görülmektedir. Doğru sınıflandırma oranı için en iyi performansı gösteren alt veri

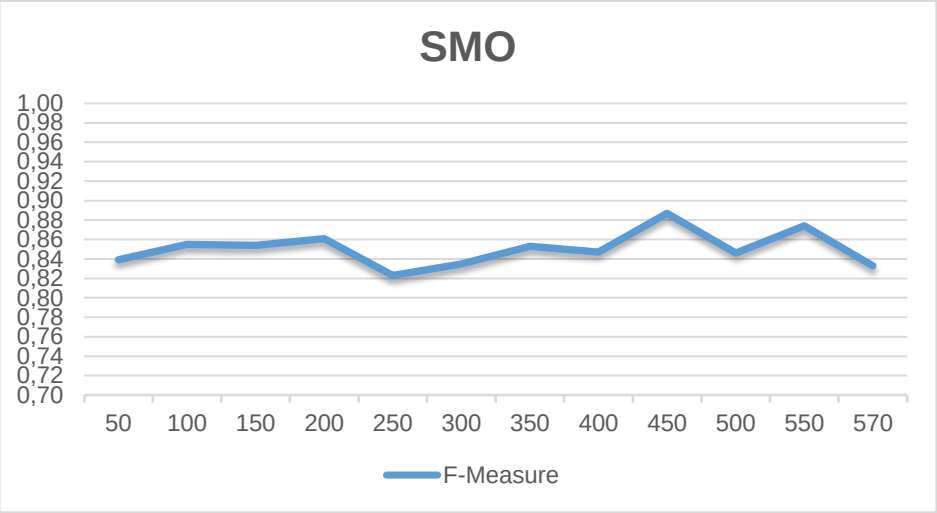
kümesinin sahip olduğu 0,29 kappa değeri, kabul edilen kappa sınırlarının altında olduğu için bu veri kümesindeki örneklerin orta derecede uyuşma içinde olduğu söylenebilir. Ancak ilgili başarı oranlarının şans eseri alındığını söylemek daha doğru olacaktır. VotedPerceptron algoritmasının alt veri kümelerinde kullanılması ile elde edilen kappa değerlerine bakarak alt veri kümelerinin neredeyse tamamının dengesiz olduğundan bahsedilebilir. Bu nedenle performans sonuçlarının şans eseri olduğu söylenebilir.

VotedPerceptron algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü örnek sayısı ilişkisinin gösterildiği mutluluk grafiği incelendiğinde ise f-ölçütü değerinin 350, 500 ve 550 örnekli alt veri kümesinde 0,86 olan en yüksek değerini aldığı görülmektedir. Mutluluk grafiğinde gösterilen f-ölçütü dalgalanmalarına bakarak deneylerde kullanılan alt veri kümelerinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığı söylenebilir. Algoritmanın bu sonuçları vermesinin nedeni olarak veri kümesindeki dengesizlik, algoritmanın kirliliğe aşırı uyum göstermesi, algoritmanın ilgili alt veri kümeleri için uyumsuz olabileceği de söylenebilir.

4.4.12. SMO Algoritması ile Doygunluk Deneyleri

Tablo 4.16 SMO Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
SMO	50	0,86	0,15	0,83	0,86	0,84	0,37
SMO	100	0,86	0,38	0,85	0,86	0,86	0,37
SMO	150	0,86	0,20	0,85	0,86	0,85	0,37
SMO	200	0,86	0,55	0,86	0,86	0,86	0,37
SMO	250	0,82	0,22	0,83	0,82	0,82	0,43
SMO	300	0,84	0,28	0,83	0,84	0,84	0,40
SMO	350	0,86	0,32	0,85	0,86	0,85	0,38
SMO	400	0,85	0,37	0,84	0,85	0,85	0,39
SMO	450	0,89	0,41	0,88	0,89	0,89	0,33
SMO	500	0,85	0,34	0,84	0,85	0,85	0,39
SMO	550	0,88	0,40	0,87	0,88	0,87	0,34
SMO	570	0,84	0,30	0,83	0,84	0,83	0,40



Şekil 4.19 SMO Algoritması Mutluluk Grafiği

570 örneğe sahip veri kümesinden, oransal olarak giderek büyüyen en az 50 örneğe sahip olmak üzere 100, 150, 200, 250, 300, 350, 400, 450, 500 ve 570 örneğe sahip olacak şekilde alt veri kümeleri seçilmiştir. Daha sonra seçilen alt veri kümeleri üzerinde SMO sınıflandırma algoritması kullanılarak deneyler yapılmıştır.

SMO algoritmasıyla 12 alt veri kümesinden elde edilen doğru sınıflandırma oranları %82 ile %89 arasında değişmektedir. En iyi performansı 450 örneğe sahip alt veri kümesinin gösterdiği ve doğru sınıflandırma başarısının %89 olduğu görülmektedir.

Örnekler arasındaki uyumu sayısal olarak ifade etmek için kullanılan kappa değerinin 0,41 olması örneklerin ekseriyetle uyumlu olduğunu ve tahminin şans eseri yapılmadığını göstermektedir. Bunun yanında genellikle en az 0,5 olarak belirlenen kabul edilebilir F-ölçütü değerinin bu veri kümesinde 0,89 olması da ilgili modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

SMO algoritmasının alt veri kümelerinde kullanılması ile elde edilen sonuçların tümüne bakıldığında eldeki kappa değerlerinin 200,450 ve 550 örneğe sahip alt veri kümeleri haricindeki diğer alt kümeler için dengesiz olduğu söylenebilir. Bu nedenle ilgili veri kümelerinden elde edilen performans sonuçlarının şans eseri olduğundan bahsedilebilir.

SMO algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü örnek sayısı ilişkisinin gösterildiği mutluluk grafiği incelendiğinde f-ölçütü değerinde gözlemlenen dalgalanmalardan ötürü örnekleme yoğunluğunun doygunluk seviyesine ulaşmadığı söylenebilir. SMO algoritmasının bu sonuçları vermesinin nedeni olarak veri

kümesindeki dengesizlik, algoritmanın kirliliğe aşırı uyum göstermesi, algoritmanın ilgili alt veri kümeleri için uyumsuz olabileceği de söylenebilir.

4.4.13. Öznitelik Dönüştürme İşleminin Doygunluğa Etkisinin İncelenmesi

Li Jing vd., ye göre veri madenciliği çalışmalarında en önemli aşamalardan birisi veri ön işlemedir. Veri ön işlem aşaması veri madenciliği çalışmalarında toplam sürenin yüzde seksenini kapsayan bir aşamadır. Bu aşama veri temizleme, kayıp değerlerin elde edilmesi ve veri dönüştürme süreçlerini içermektedir [66].

Veri ön işlem süreçlerinden biri olan veri dönüştürme işlemi, algoritmaların sınıflandırma performanslarını daha iyi hale getirmek için örneklerin sahip olduğu özniteliklerin değerlerinin dönüştürülmesi anlamına gelir. Örneğin akademik başarı ile ilgili olan bir özniteliğin aldığı zayıf, orta, iyi, pekiyi gibi değerler yerine başarılı ve başarısız olarak dönüştürülmesi algoritmaların sınıflandırma performansını arttırmaya yönelik yapılmış bir dönüşüm olacaktır. Veri dönüştürme işleminin sonucunda elde edilen değer, model için anlam ifade etmelidir [67].

Mevcut veriler içerisinden uygun olan öznitelikler seçilerek veri dönüştürme işlemine tabi tutulmuştur. Veri kümesinde 36 öznitelikten uygun olan 14 öznitelik dönüştürülmüştür. Aşağıdaki tabloda dönüştürülen özniteliklerin veri kümesindeki sıra numarası, adı, sahip olabileceği değerler ve yeni değerleri verilmiştir.

Tablo 4.17.Öznitelik Dönüşüm Tablosu

Sn	Öznitelik	Eski Değerler	Yeni Değerler
5	nerede_yasiyorsun	{ailemle,yurtta,akrabada,hicbiri,bos}	ailemle,diger
20	okulda_vakit_gecirme	{hicbirzaman,bazen,siksik,herzaman,bos}	evet,hayir
22	ders_calisma_vakti	{hicbirzaman,bazen,siksik,herzaman,bos}	evet,hayir
23	odev_yapıyormusun	{hicbirzaman,bazen,siksik,herzaman,bos}	evet,hayir
24	arakadaslar_ile_vakit_gecirme	{hicbirzaman,bazen,siksik,herzaman,bos}	evet,hayir
25	yaz_tatili	{hicbirzaman,bazen,siksik,herzaman,bos}	evet,hayir
26	kis_tatili	{hicbirzaman,bazen,siksik,herzaman,bos}	evet,hayir
27	kitap_okumayi_seviyormusun	{hicbirzaman,bazen,siksik,herzaman,bos}	evet,hayir
28	kac_saat_bilgisayar_kullanıyorsun	{kullanmiyorum,1_2_saat,2_4_saat,4_saaten_fazla,bos}	hayir,az,cok
29	kac_saat_tv	{kullanmiyorum,1_2_saat,2_4_saat,4_saaten_fazla,bos}	hayir,az,cok

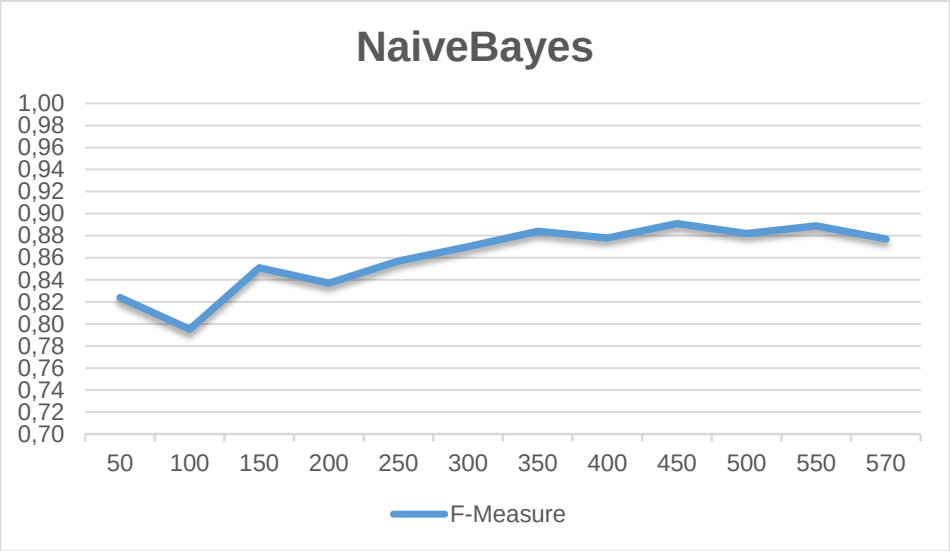
31	sira_arakadasindan_memnunumusun	{memnundegilim,bazenmemnunum,geneldememnunum,cok memnunum,bos}	evet,hayir
32	okul_etkinliginde_gorev	{hicbirzaman,bazen,siksik,herzaman,bos}	evet,hayir
33	uyku_saati	{saat8de,saat9da,saat10da,saat11veyadahagec,bos}	erken,gec
34	ogretmen_isimlerini_biliyormusun	{hicbirini_bilmiyorum,birkacini_biliyorum,cogunun_ismini_ biliyorum,hepsinin_ismini_biliyorum,bos}	evet,hayir

Dönüşüm sonrasında geçerli veri kümesi kullanılarak, en küçük veri kümesinin 50 örnekten oluştuğu daha sonra 100 örneğe sahip veri seti ve 50 örnek arttırarak oluşturulan 150, 200, ….550 diyerek giden ve en son 570 örnekten oluşan alt veri kümeleri seçildi.

Seçilen veri kümelerinde sınıflandırma algoritmaları kullanılarak elde edilen sonuçlar tablolara aktarılmıştır. Algoritmalardan elde edilen f-ölçütü ile örnek sayıları kullanılarak mutluluk grafiği oluşturulmaya çalışılmıştır. Ancak kullanılan algoritmalardan sadece Naïve Bayes sınıflandırma algoritmasının sahip olduğu değerler mutluluk grafiği oluşturmaya yakın bulunmuştur. Aşağıda Naïve Bayes algoritmasının oniki adet alt veri kümesi üzerinde çalıştırılmasından elde edilen sonuçlar, mutluluk grafiği ve sonuç analizi yer almaktadır.

Tablo 4.18 Naïve Bayes Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
bayes.Naïve Bayes	50	0,88	0,00	0,77	0,88	0,82	0,34
bayes.Naïve Bayes	100	0,86	0,00	0,74	0,86	0,80	0,36
bayes.Naïve Bayes	150	0,89	0,07	0,84	0,89	0,85	0,31
bayes.Naïve Bayes	200	0,86	0,42	0,84	0,86	0,84	0,35
bayes.Naïve Bayes	250	0,88	0,25	0,86	0,88	0,86	0,31
bayes.Naïve Bayes	300	0,89	0,40	0,87	0,89	0,87	0,31
bayes.Naïve Bayes	350	0,90	0,43	0,88	0,90	0,88	0,29
bayes.Naïve Bayes	400	0,89	0,47	0,88	0,89	0,88	0,31
bayes.Naïve Bayes	450	0,90	0,41	0,89	0,90	0,89	0,28
bayes.Naïve Bayes	500	0,89	0,49	0,88	0,89	0,88	0,30
bayes.Naïve Bayes	550	0,90	0,47	0,89	0,90	0,89	0,29
bayes.Naïve Bayes	570	0,88	0,50	0,88	0,88	0,88	0,31



Şekil 4.20 Naïve Bayes Algoritması Mutluluk Grafiği

14 özniteliği dönüştürülmüş 570 örneğe sahip veri kümesinden, en az 50 örneğe sahip olmak üzere 100, 150, 200, 250, 300, 350, 400, 450, 500 ve 570 örneğe sahip olan alt veri kümeleri seçilmiştir. Daha sonra seçilen alt veri kümeleri üzerinde Naïve Bayes sınıflandırma algoritması kullanılarak deneyler yapılmıştır.

Deney sonuçlarına bakıldığında performansı en yüksek veri kümesinin 350, 450 ve 550 örneğe sahip alt veri kümeleri olduğu görülmektedir. Bu üç veri kümesinin doğru sınıflandırma başarısının %90 olduğu görülmektedir. Hangi veri kümesinin en başarılı sonuçları elde ettiğini belirlemek için ikinci bir parametre olan kappa değerinin dikkate alınması en başarılı veri kümesinin bulunması için gerekli olacaktır. 350 örnekli veri kümesinin 0,43 iken, 450 örnekli veri kümesinin kappa değeri 0,41 dir. 550 örnekli veri kümesi 0,47 kappa değeri ile üçü arasında en yüksek değere sahiptir. Bu nedenle doğru sınıflandırma performansları eşit olan bu üç veri kümesi arasında kappa değeri en yüksek olan 550 örneğe sahip veri kümesi en başarılı veri kümesidir.

Kappa değerinin örnekler arasındaki uyumu sayısal olarak ifade etmekte kullanıldığı bilinmektedir. 550 örneğe sahip alt veri kümesinde 0,47 olduğu ve bu değerin örneklerin ekseriyetle uyumlu olduğunu ve tahminin şans eseri yapılmadığını göstermektedir. Bunun yanında genellikle en az 0,5 olarak belirlenen kabul edilebilir F-ölçütü değerinin bu veri kümesinde 0,89 olmasının da ilgili modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

Naïve Bayes sınıflandırma algoritmasının alt veri kümelerinde kullanılması ile elde edilen f-ölçütü – örnek sayısı değerleri kullanılarak bir grafik elde edilmiştir. Grafikte f-ölçütü değerinin 50 örnekli, 100 örnekli, 150 örnekli ve 250 örnekli alt veri kümelerinden alınan sırasıyla 0,82 , 0,80 , 0,85 ve 0,86 şeklinde ilerleyen f-ölçütü değerleri görülmektedir. F-ölçütü sonuçlarının yüksek olmasının tesadüfi olup olmadığının anlaşılması için bu alt veri kümelerinin kappa değerlerini incelemek gerekecektir. 50, 100 ve 150 örnekli alt veri kümelerinin kappa değerleri sırasıyla 0, 0 ve 0,07 olması bu üç alt veri kümesinden elde edilen performans sonuçlarının şans eseri olduğunu göstermektedir. 200 örnekli alt veri kümesinin f-ölçütü değerinin ani bir şekilde yükselme ve 250 örnekli veri setine gelindiğinde tekrar düşme eğiliminde olmasının bu alt veri kümelerinde kirlilik bulunmasıyla açıklanabilir. Grafik takip edildiğinde 250 örneğe sahip olan alt veri kümesinde de kappa değerinin 0,25 olmasından dolayı elde edilen performans değerinin şans eseri olduğu söylenebilir. 250 örnekli alt veri kümesinden sonra örnek sayısının artmasıyla f-ölçütü değerlerinin 0,882 ile 0,889 arasında değiştiği görülüyor. Bu değişim deneylerde kullanılan verinin kirliliğe sahip olmasına rağmen örnekleme yoğunluğunun doygunluk seviyesine ulaşmış olduğu sonucunun çıkarılmasına olanak vermektedir.

4.4.14. Öznitelik Eksiltmenin Doygunluğa Etkisinin İncelenmesi

Veri kümesindeki örneklerin sahip olduğu özniteliklerde yapılacak olan değişikliğin performansa bir etkisi olup olmadığının test edilmesi için özniteliklerin sayısında eksiltme yapılabilir. Bu işlem için, veri kümelerinin sahip olduğu her bir özniteliğin, sınıflandırıcı değerine ulaşma noktasında ne kadar enformasyon sağladığını bulmak için kullanılan InfoGainAttributeEval algoritmasından faydalanılabilir. Çalışmada InfoGainAttributeEval 570 örneğe sahip mevcut veri kümesinde kullanılarak özniteliklerin sınıflandırma değerine etkisinin listesi elde edilmiştir. Bu listeden değeri 0,02’ nin altında olan öznitelikler tüm örneklerden çıkarılarak örneklerin 36 yerine 12 adet öznitelikle sınıflandırılması sağlanmıştır. Aşağıda InfoGainAttributeEval algoritmasının 570 örnekli alt veri kümesi üzerinde çalıştırılması sonucunda elde edilen sıralı liste gösterilmiştir.

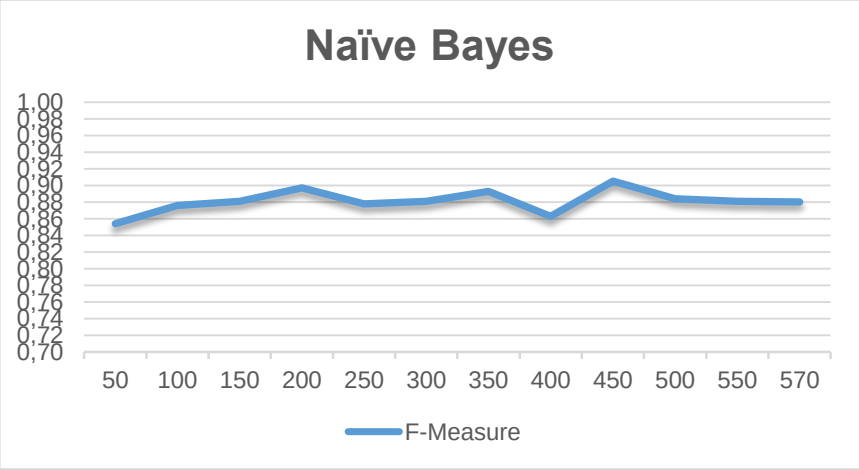
```
Ranked attributes:
 0.1603188   35 ontest
 0.0830433    2 gelir_duzeyi
 0.0596522   13 babanin_egitim_durumu
 0.0373159   23 odev_yapÄ±yormusun
 0.0305591   25 yaz_tatili
 0.0287897   10 annenin_egitim_durumu
 0.0283013    1 probleme_dayali
 0.0244391   33 uyku_saati
 0.0240476   14 kac_kardessiniz
 0.0212228   19 ulasim
 0.0200178   16 odevlerine_kim_yardimci_oluyor
 0.0187013   24 arakadaslar_ile_vakit_gecirme
 0.0184602    7 odan_varmi
 0.0161956   22 ders_calisma_vakti
 0.0161489   18 adresini_biliyormusun
 0.0136337   26 kis_tatili
 0.0121652   29 kac_saat_tv
 0.0117655   27 kitap_okumayi_seviyormusun
 0.0106186    6 kiminle_yasiyorsun
 0.0097478   34 ogretmen_isimlerini_biliyormusun
 0.008591    32 okul_etkinliginde_gorev
 0.0082992   20 okulda_vakit_gecirme
 0.0080182   15 engelli_akraba
 0.0076293   17 telefon_numarasini_biliyormusun
 0.0068317    4 yas
 0.006038    21 en_coksevdigin_ders
 0.0057269   12 baban_calisiyormu
 0.0053629    5 nerede_yasiyorsun
 0.0048828   28 kac_saat_bilgisayar_kullanÄ±yorsun
 0.0024607   11 baban_yasiyormu
 0.0019444   31 sira_arakadasindan_memnunumusun
 0.0012625   30 evcil_hayvan
 0.0007981    8 annen_yasiyormu
 0.0005641    3 cinsiyet
 0.00009      9 annen_calisiyormu
```

Şekil 4.21 Örnekli Veri kümesinin InfoGainAttributeEval Algoritması Sonuç Listesi

12 özniteliğe sahip 570 örnekli veri kümesinden, en az 50 örneğe sahip olmak üzere 100, 150, 200, 250, 300, 350, 400, 450, 500 ve 570 örneğe sahip olan alt veri kümeleri seçilmiştir. Çalışmada kullanılan LWL, J48, JRIP, Part, LMT, Baggıng, Random Forest, IBK, MultiLayer Perceptron, Voted Perceptron, SMO, Naïve Bayes algoritmaları ile seçilen on iki adet veri kümesi üzerinde deneyler yapılmıştır. Deney sonuçlarına bakıldığında Naïve Bayes ve Voted Perceptron algoritmalarında verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmış olduğu sonucu çıkarılabilmektedir. Aşağıda bu sonuçlardan bahsedilmiştir.

Tablo 4.19 Naïve Bayes Algoritması Doygunluk Analizi Sonuçları

Algoritma	Instances	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
bayes.Naïve Bayes	50	0,86	0,29	0,85	0,86	0,85	0,32
bayes.Naïve Bayes	100	0,88	0,47	0,87	0,88	0,88	0,33
bayes.Naïve Bayes	150	0,89	0,35	0,88	0,89	0,88	0,29
bayes.Naïve Bayes	200	0,90	0,66	0,90	0,90	0,90	0,30
bayes.Naïve Bayes	250	0,88	0,43	0,88	0,88	0,88	0,29
bayes.Naïve Bayes	300	0,88	0,50	0,88	0,88	0,88	0,31
bayes.Naïve Bayes	350	0,89	0,51	0,89	0,89	0,89	0,29
bayes.Naïve Bayes	400	0,86	0,46	0,87	0,86	0,86	0,31
bayes.Naïve Bayes	450	0,91	0,52	0,90	0,91	0,91	0,27
bayes.Naïve Bayes	500	0,88	0,52	0,88	0,88	0,88	0,30
bayes.Naïve Bayes	550	0,88	0,47	0,88	0,88	0,88	0,29
bayes.Naïve Bayes	570	0,88	0,53	0,88	0,88	0,88	0,30



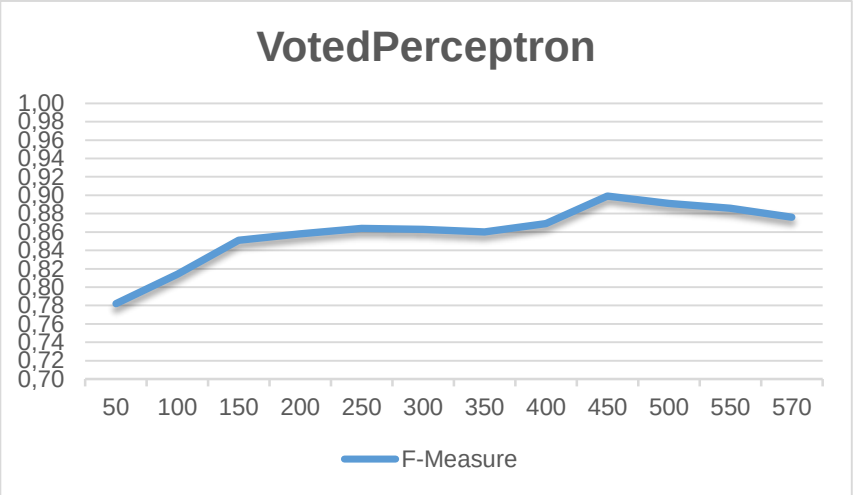
Şekil 4.22 Naïve Bayes Algoritması Mutluluk Grafiği

NavieBayes algoritmasının ilgili alt veri kümelerinde gerçekleştirilen deney sonuçlarına bakıldığında doğru sınıflandırma başarısı %91 olan ve bu nedenle en yüksek performansa sahip veri kümesinin 450 örneğe sahip alt veri kümesi olduğu görülmektedir. Bu alt kümenin örnekleri arasındaki uyumu değerlendirebilmek için kappa değerinden yararlanılmıştır. 450 örnekli alt veri kümesi için kappa değeri 0,52' dir. Kappa değerinin 0,41 ile 0,60 arasında olması örneklerin birbiriyle ekseriyetle uyumlu olduğunu ve tahminin şans eseri yapılmadığını göstermektedir. F-ölçütü değerinin bu veri kümesinde 0,91 olması da ilgili modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

NavieBayes algoritmasının deney sonuçlarından elde edilen f-ölçütü ile örnek sayısı kullanılarak çizilen mutluluk grafiği verilmiştir. 50 örneğe sahip veri kümesinde f-ölçütü değerinin 0,85 daha sonraki 100 örneğe sahip veri kümesinden elde edilen f-ölçütü değerinin 0,88 olarak devam etmiştir. Bu noktadan sonra 570 örnekli veri kümesine kadar inişli çıkışlı bir eğilim görülmektedir. 200, 350, 400, 450 örnekli veri kümelerinde dalgalanmalar gözlenmesinin sebebi veri kümelerinde bulunan kirlilikle açıklanabilir. Bu nedenle ilgili mutluluk grafiğindeki doygunluk başlangıç noktasının ikinci veri kümesinde başladığı söylenebilir. Buradaki 100 örnekli veri kümesindeki doğru sınıflandırma oranı 0,88 kappa değeri 0,47 ve f-ölçütü değeri 0,88 dir.

Tablo 4.20 VotedPerceptron Algoritması Doygunluk Analizi Sonuçları

Algoritma	**Instances**	**Correctly Classified**	**Kappa**	**Precision**	**Recall**	**F-Measure**	**RMSE**
VotedPerceptron	50	0,80	-0,11	0,77	0,80	0,78	0,44
VotedPerceptron	100	0,83	0,17	0,80	0,83	0,81	0,41
VotedPerceptron	150	0,89	0,07	0,84	0,89	0,85	0,33
VotedPerceptron	200	0,87	0,50	0,86	0,87	0,86	0,36
VotedPerceptron	250	0,88	0,31	0,86	0,88	0,86	0,34
VotedPerceptron	300	0,88	0,36	0,86	0,88	0,86	0,34
VotedPerceptron	350	0,88	0,32	0,86	0,88	0,86	0,35
VotedPerceptron	400	0,88	0,44	0,87	0,88	0,87	0,35
VotedPerceptron	450	0,91	0,44	0,90	0,91	0,90	0,30
VotedPerceptron	500	0,90	0,52	0,89	0,90	0,89	0,32
VotedPerceptron	550	0,89	0,45	0,88	0,90	0,89	0,32
VotedPerceptron	570	0,89	0,46	0,87	0,89	0,88	0,34



Şekil 4.23 VotedPerceptron Algoritması Mutluluk Grafiği

Voted Perceptron algoritmasının seçili olan alt veri kümelerinde gerçekleştirdiği deney sonuçlarına bakıldığında en yüksek performansın 450 örneğe sahip alt veri kümesinden elde edildiği görülmektedir. Bu sonuçlara göre veri kümesinin doğru sınıflandırma başarısının %91 olduğu görülmektedir. Tahminin şans eseri olup olmadığını sayısal olarak ifade etmek için kullanılan kappa değerinin 0,44 olması da örneklerin ekseriyetle uyumlu olduğunu ve tahminin şans eseri yapılmadığını göstermektedir. Bunun yanında F-ölçütü değerinin ilgili veri kümesinde 0,90 olması da ilgili modelin sınıflandırma tanısı için yeterli olduğunu göstermektedir.

Voted Perceptron algoritmasının oluşturduğu mutluluk grafiğine bakıldığında f-ölçütünün 500 örneğe sahip veri kümesine kadar yükseldiğinin burada yakaladığı 0,89 değerini 550 örnekli veri kümesine kadar koruduğu daha sonra azalma eğilimine girdiği görülmektedir. Grafik daha yakından incelendiğinde doygunluk başlangıç noktasının 500 örnekli veri kümesi olduğu söylenebilir. Voted Perceptron algoritmasının 500 örnekli alt veri kümesinde aldığı doğru sınıflandırma oranı 0,90 kappa değeri 0,52 ve f-ölçütü değeri 0,89 dur.

İki algoritmadan elde edilen sonuçlar karşılaştırıldığında en yüksek performans değerinin iki algoritma içinde 450 örneğe sahip olan alt veri kümesinden elde edildiği görülmektedir. İki algoritmanın da sahip olduğu en yüksek performans değeri 450 örneğe sahip alt veri kümesinde gerçekleşmiştir.

4.4.15. Doygunluk Deneyleri Analiz Sonuçları

Doygunluk çalışması, veri kümesi oluşturma, önişlem ve analiz adımlarını içermektedir. Çalışmada kullanılacak veri kümesinin oluşturulması aşamasında öğrencilerin test ve anket uygulamasını yaparak optik formları doldurmaları istenmiş uygulama sonrasında optik formlar ile alınan öğrenci yanıtları geliştirilen program yardımıyla veri tabanına aktarılmıştır. Daha sonra veri tabanında bulunan örnekler üzerinde sınıflandırma başarısını olumsuz etkileyecek olası kirlilik durumlarını ortadan kaldırmak için geliştirilen program kullanılarak bazı önişlem basamakları gerçekleştirilmiştir. Buna göre elde edilen örneklerden anketi yanıtlayıp testi yanıtlamayanların, testi yanıtlayıp anketi yanıtlamayanların, anket ve test cevaplarını boş bırakanların, annesi vefat etmesine rağmen annem ile kalıyorum veya babası vefat etmesine rağmen babamla kalıyorum seçeneklerini işaretleyen kayıtların veri tabanından çıkarılması sağlanmıştır. Bu ön işlem süreci sonrasında 570 örneğe sahip veri kümesi elde edilmiştir.

Doygunluk deneyleri için 570 örnekli üç farklı veri kümesi oluşturulmuştur. Bunlardan ilki yukarda bahsedilen önişlem sonrasında elde edilen veri kümesidir. İkincisi ilk duruma konu olan veri kümesi üzerinde öznitelik dönüştürmesi uygulandıktan sonra elde edilen veri kümesidir. Üçüncüsü ise ilk elde edilen veri kümesinden öznitelik eksilttikten sonra ortaya çıkan veri kümesidir. Bu üç veri kümesi üzerinde sınıflandırma algoritmaları kullanılarak deneyler yapılmıştır. Aşağıda kısaca kullanılan üç veri kümesinden elde edilen sonuçlar hakkında bilgiler verilmiştir.

İlk veri kümesinde bulunan 570 örnekten, en az 50 örneğe sahip olmak üzere 100, 150, 200, 250, 300, 350, 400, 450, 500 ve 570 örneğe sahip olan alt veri kümeleri seçilmiştir. Bu alt veri kümeleri üzerinde çalıştırılan LWL, J48, JRIP, Part, LMT, Baggıng, Random Forest, IBK, MultiLayer Perceptron, Voted Perceptron, SMO, Naïve Bayes sonucunda elde edilen değerler bir tabloya aktarılmış, bunun yanında algoritmaların elde ettikleri f-ölçütü değerleri ile alt veri kümelerindeki örnek sayılarının yer aldığı mutluluk grafikleri çizilmiştir. Grafikler incelendiğinde sadece Naïve Bayes algoritmasında verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmış olduğu sonucu çıkarılmaktadır. Naïve Bayes mutluluk grafiğine bakıldığında doygunluk başlangıç noktasının 300 örnekli alt veri kümesi olduğu ve bu alt veri kümesi için doğru

sınıflandırma oranının 0,89 kappa değerinin 0,52 ve f-ölçütü değerinin 0,89 olduğu söylenebilir.

İkinci veri kümesinin elde edilmesi noktasında veri kümesinin sahip olduğu her bir özniteliğin, sınıflandırıcı değerine ulaşma noktasında ne kadar enformasyon sağladığını bulmak için kullanılan InfoGainAttributeEval algoritmasından faydalanılmıştır. InfoGainAttributeEval algoritmasının verdiği sonuçlar dikkate alınarak veri setinden özniteliklerin etkisinin 0,02 değerinin altında olduğu öznitelikler çıkarılarak her bir örneğin 12 özniteliğe sahip olduğu 570 örnekli veri kümesi oluşturulmuştur. İşlem sonrasında veri kümesinden en küçüğü 50 örnekli olan ve 50 şer artarak 100, 150, 200, 250, 300, 350, 400, 450, 500 ve en son 570 örnekli toplam on iki alt veri kümesi seçilmiştir. Bu alt veri kümeleri üzerinde çalıştırılan LWL, J48, JRIP, Part, LMT, Baggıng, Random Forest, IBK, MultiLayer Perceptron, Voted Perceptron, SMO, Naïve Bayes sonucunda elde edilen değerler bir tabloya aktarılmış ve mutluluk grafikleri çizilmiştir. Grafikler incelendiğinde sadece Naïve Bayes algoritmasında verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmış olduğu sonucu çıkarılmaktadır. Naïve Bayes mutluluk grafiğine bakıldığında doygunluk başlangıç noktasının 350 örnekli alt veri kümesi olduğu ve bu veri alt kümesi için doğru sınıflandırma oranının 0,90 kappa değerinin 0,43 ve f-ölçütü değerinin 0,88 olduğu söylenebilir.

Üçüncü yani ilk duruma konu olan veri kümesinden öznitelik eksilttikten sonra ortaya çıkan veri kümesinden 50, 100, 150, 200, 250, 300, 350, 400, 450, 500 ve 570 örnekli toplam on iki alt veri kümesi seçilmiştir. Elde edilen alt veri kümeleri LWL, J48, JRIP, Part, LMT, Baggıng, Random Forest, IBK, MultiLayer Perceptron, Voted Perceptron, SMO, Naïve Bayes algoritmaları ile çalıştırılmıştır. Bu algoritmalara ait mutluluk grafikleri ve deney sonuçları incelendiğinde sadece Naïve Bayes ve Voted Perceptron algoritmalarında verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmış olduğu sonucu çıkarılabilmektedir. NavieBayes algoritmasının mutluluk grafiğine bakıldığında doygunluk başlangıç noktasının daha 100 örnekli ikinci veri kümesinde başladığı söylenebilir. Buradaki doğru sınıflandırma oranı 0,88 kappa değeri 0,47 ve f-ölçütü değeri 0,88 dir. Daha sonra 200, 350, 400, 450 örnekli veri kümelerine ait sonuçlarda f-ölçütü çizgisi dalgalı bir görünüm almıştır. Bu alt veri kümeleri için kirlilikten söz edilebilir. Voted Perceptron algoritmasının mutluluk grafiğine bakıldığında f-ölçütünün 500 örneğe sahip veri kümesine kadar yükseldiğinin burada yakaladığı 0,89

değerini 570 örnekli veri kümesine kadar koruduğu ve bu noktada azalma eğilimine girdiği görülmektedir. Voted perceptron algoritmasının mutluluk grafiği incelendiğinde doygunluk başlangıç noktasının 500 örnekli veri kümesi olduğu söylenebilir. Voted Perceptron algoritmasının 500 örnekli alt veri kümesinde aldığı doğru sınıflandırma oranı 0,90 kappa değeri 0,52 ve f-ölçütü değeri 0,89 dur.

Yapılan tüm deneyler göstermiştir ki Naïve Bayes ve Voted Perceptron algoritmalarının tutarlı bir şekilde mutluluk çizgisi sunması kullanılan veri kümesinin aslında doygun olduğunu göstermektedir. Ancak diğer algoritmaların muhtemelen bir özelliğinden dolayı veya bu algoritmaların bu veri kümesindeki veriler için aşırı güçlü olmasından dolayı kirliliğe çok fazla uyum göstererek mutluluk çizgisi elde edilemediği düşünülmektedir.

4.5. Veri Kümesi Karmaşıklık Deneyleri

Önceki bölümde veri kümesinin sahip olduğu verilerin yeterliliği konusunda yapılan deneyler sonucunda Naïve bayes ve voted perceptron algoritmalarının örnek sayısı ile f-ölçütü ilişkisini veren grafiklerde mutluluk grafikleri (happy graphs) elde edilebilmiştir. Bu grafiklerin elde edilmiş olması, mevcut veri kümesinin doygunluk düzeyi hakkında fikir vermekte ve bunun yanında verinin yeterli olduğunu da göstermektedir. Karmaşıklık analizi verinin kendi içinde ne kadar karmaşık ilişkiler barındırdığını çözümlemeye çalıştığı için verinin yetersizliğinden kaynaklı performans problemleri karmaşıklıkla ilgili problemleri gölgeleyecekti.

Karmaşıklık analizi için hangi deneylerin yapılacağının belirlenmesinde Kılıçaslan vd., 2009 çalışmasından yararlanılmıştır. Kılıçaslan vd., 2009 çalışmasında yapılan deneyler mevcut veri kümesine uygulanmıştır. Kılıçaslan vd., 2009 kullandıkları veri kümesinin karmaşıklığını belirlemek için beş sınıflandırma algoritmasından yararlanmıştır. Bu kaynak çalışmada IBK, SMO, Voted Perceptron, J48 ve Naïve Bayes algoritmaları kullanılmıştır. Kaynak çalışmada her algoritma için özel olarak belirlenen parametreler bu çalışmada da aynı şekilde kullanılmıştır.

Aşağıda bu tez çalışması için oluşturulmuş anket - test öznitelik ve bunlara ait değerleri barındıran eğitim alanlı veri kümesinde yukarıda bahsi geçen kaynak çalışmada kullanılan algoritmalar çalıştırılmıştır. Elde edilecek sonuçlar mevcut veri kümesinin

karmaşıklık düzeyini belirlemek amacıyla kullanılmıştır. Sonuçlar ayrıca Kılıçaslan vd., 2009 'nin elde ettikleri sonuçlar ile karşılaştırılarak yorumlanmıştır.

4.5.1. Voted Perceptron Karmaşıklık Deneyleri

Tablo 4.21 VotedPerceptron Algoritması Karmaşıklık Deneyleri Sonuçları

Exponent	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
1	0,89	0,46	0,87	0,89	0,88	0,34
2	0,88	0,46	0,87	0,88	0,87	0,35
3	0,87	0,46	0,87	0,87	0,87	0,36
4	0,86	0,41	0,85	0,87	0,86	0,37
5	0,86	0,41	0,85	0,86	0,86	0,37

Voted perceptron algoritması verinin lineer olduğu durumlarda, başarılı sonuçlar veren bir algoritmadır. Bu durumu sağlayabilmek için üs(exponent) özelliği değerinin 1 yani lineer olması sağlanmalıdır. Veri kümesinde voted perceptron algoritmasının farklı üs değerleri ile deneyler yapılmış ve bu deneyler bir tabloda toplanmıştır. Tablo incelendiğinde üs değeri arttıkça doğru sınıflandırma oranının düştüğü görülmektedir. Aynı şekilde örneklerin kendi aralarındaki ilişkiyi veren kappa değeri ve modelin tanı için yeterli olduğunu gösteren f-ölçütü değeri de üs değerinin artmasıyla birlikte azalmaktadır. En iyi değerler üssü 1 alarak elde edilmiştir. Bu durumda doğru sınıflandırma oranı 0,89, kappa değeri 0,46 ve f-ölçütü 0,88 dir. Bu durum sınıflandırma tahminin şans eseri yapılmadığını ve modelin tanı için yeterli olduğunu göstermektedir. Üs değerinin artması voted perceptron algoritmasının doğrusallıktan uzaklaşarak karmaşıklaşması anlamına gelmektedir. Tablodan üs değeri ile sınıflandırma başarısının ters orantılı olduğu rahatlıkla görülebilmektedir. Bu durumun sebebi veri kümesinin karmaşık olmaması, lineere yakın olması ile açıklanabilir.

Veri kümesinde voted perceptron algoritmasının farklı üs değerleri ile yapılan deneyleri incelendiğinde elde edilen doğru sınıflandırma oranında, kappa değerinde ve f-ölçütü değerinde de üs değerindeki artışa ters olarak bir değer azalması görülmektedir. Bu da verinin lineer olduğunun kanıtı olarak söylenebilir. Ayrıca üs değerinin artması algoritmanın ifade gücünün artması anlamına da gelmektedir. Algoritma üs değeri 4 ve 5 alındığında elde edilen değerlerin sırasıyla doğru sınıflandırma oranı 0,86, kappa değerinin 0,41 ve f-ölçütü değerinin 0,86 olduğu görülmektedir. Üs değerinin 4 ve 5

alınmasıyla kappa değeri aniden düşmüştür. F-ölçütü değerinin düşmesi de voted perceptron algoritmasının bu üs değerlerinde aşırı uyum (overfitting) olduğunu desteklemektedir. Bu aşırı öğrenme durumu ifade gücü arttırılmış bir algoritmanın lineer bir veri ile karşılaşması durumunda ortaya çıkabilecek istenmeyen bir durumdur. Aşırı öğrenmenin ortaya çıkması kullanılan verinin lineer olduğunun başka bir ispatı olarak ifade edilebilir.

Kılıçaslan vd., 2009 yaptığı çalışmada, ilk deneyde Voted perceptron algoritmasının üs değerini 1 olarak almış ve doğruluk oranı 0,79, kappa değeri 0,52 ve f-ölçütü 0,68 sonuçlarını elde etmiştir. İkinci deneyde voted perceptron algoritmasının üs değerini 2 olarak almış ve doğru sınıflandırma oranı 0,81, kappa değeri 0,57 ve f-ölçütü olarak 0,71 sonuçları alınmıştır. Deneyler sonucunda üs değerlerinin artmasının kappa, f-ölçütü ve doğru sınıflandırma değerine olumlu olarak yansıdığı vurgulanmıştır. Bu sonuçlar ilgili çalışmada kullanılan veri kümesinin karmaşık olduğunun bir ispatı olmuştur. Bu çalışmada kullanılan veri kümesinin aynı deneylerde farklı yönde eğilim göstermesi verinin lineer olduğunun ispatı olarak sunulabilir.

4.5.2. IBK Karmaşıklık Deneyleri

Tablo 4.22 IBK Algoritması Karmaşıklık Deneyleri Sonuçları

K Değeri	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
1	0,87	0,37	0,85	0,87	0,86	0,35
11	0,89	0,36	0,88	0,89	0,86	0,29
15	0,88	0,26	0,88	0,88	0,84	0,29
20	0,88	0,25	0,88	0,88	0,84	0,29

IBK yakın komşu algoritması doğrusal olmayan bir algoritmadır ve sınıflandırma açısından yüksek kapasiteye sahiptir. Sahip olduğu bu güçle sınıf belirsizliği olmadığı sürece mükemmel sınıflandırma yapabilmektedir. IBK algoritmasıyla dört deney yapılmıştır ve sonuçlar bir tablo ile sunulmuştur. Bu deneylerden ilkinde k değerine 1 verilmiş ve algoritmanın çalıştırılması sonucunda doğru sınıflandırma oranı olarak 0,87 kappa değeri olarak 0,37 ve f-ölçütü olarak 0,86 değerleri elde edilmiştir. İkinci deneyde k değeri olarak 11 verilmiş ve algoritma sonucunda doğru sınıflandırma oranı 0,89 kappa değeri 0,36 ve f-ölçütü 0,86 değerleri elde edilmiştir. Üçüncü deneyde k değeri 15 olarak

belirtilmiş algoritma çalıştırıldığında doğru sınıflandırma oranı 0,88 kappa değeri 0,26 f-ölçütü değeri 0,84 olmuştur. Dördüncü deneyde k değeri 20 olarak belirlenmiş ve algoritma çalıştırıldığında doğru sınıflandırma oranı 0,88 kappa 0,25 ve f-ölçütü 0,84 değerleri alınmıştır.

IBK algoritmasında belirlenen k değerinin artması algoritmanın ifade gücünü arttırmaktadır. Ancak doğrusal olan veri kümelerinde deneyler yapılırken k değeri doğru seçilmelidir aksi takdirde aşırı uyum denilen durum gerçekleşir. Tabloda k değerinin 15 ve 20 olarak kullanıldığı deneylerde f-ölçütünün 0,84 ve kappa değerinin 0,26 - 0,25 olması modelin eğitim kümesine aşırı uyum gösterdiğini yani overfitting olduğunu göstermektedir.

Kılıçaslan vd., 2009 yaptığı çalışmaya bakıldığında, yapılan IBK deneylerinde k değerinde yapılan artış kappa, f-ölçütü ve doğru sınıflandırma değerine olumlu olarak yansımıştır. Şöyle ki ilgili veri kümesinde IBK algoritmasının k=1 değeri için elde ettiği sonuç doğru sınıflandırma oranı olarak 0,75 kappa değeri için 0,48 ve f-ölçütü 0,67 dir. IBK algoritmasının k=11 değeri için elde ettiği sonuç doğru sınıflandırma oranı olarak 0,80 kappa olarak 0,55 ve f-ölçütü olarak 0,72 dir. K değerinin 1 olduğu deneyde alınan düşük sonuçlar için algoritmanın ifade gücünü arttırması nedeniyle gürültülü verilerin aşırı uyumu ortaya çıkardığı söylenebilir.

4.5.3. SMO Karmaşıklık Deneyleri

Tablo 4.23 SMO Algoritması Karmaşıklık Deneyleri Sonuçları

Kernel	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
Linear Kernel	0,88	0,47	0,87	0,88	0,87	0,35
RBF Kernel	0,85	0,00	0,73	0,85	0,79	0,38

Destek karar makineleri (SVM), yüksek boyutlu verilerde sıkça kullanılan bir sınıflandırma metodudur. SVM doğrusal problemleri çözebildiği gibi doğrusal olmayan problemleri de çözebilir. Bunu doğrusal olmayan dönüşüm ile giriş aralığını doğrusal olmayan aralıkla eşleştirerek yapar. Bu dönüşüm için (Polynomial kernel / Linear Kernel) doğrusal çekirdek veya Radyal Tabanlı çekirdek (Radial Basis Functions Kernel) gibi değişik çekirdekler sıklıkla kullanılmaktadır. Bu çalışmada veri kümesini farklı

çekirdeklerle sınıflandırma amacıyla SVM kullanılmıştır. SVM aracı Weka da SMO aracı ile kullanılmıştır [68].

Kılıçaslan vd., 2009 yapmış oldukları destek vektör deneylerinde veri kümesinde doğrusal ve radyal tabanlı çekirdek fonksiyonları kullanmıştır. Deneylerde elde edilen sonuçlara bakıldığında doğrusal çekirdeğin kullanıldığı deneyde elde edilen doğru sınıflandırma oranı 0,77 kappa değeri 0,47 ve f-ölçütü değeri 0,64 tür. Radyal çekirdeğin kullanıldığı deneylerde elde edilen değer doğru sınıflandırma oranı için 0,82 kappa değeri 0,60 ve f-ölçütü değeri 0,74 olmuştur. Burada çekirdek değiştirilerek yapılan güç arttırma işlemi ile algoritmanın ifade gücü arttırılmıştır. Yapılan bu değişikliğin algoritmanın performansını olumlu yönde etkilemiş olması deneyde kullanılan verinin karmaşık olduğunun da bir göstergesi olarak söylenebilir.

Kılıçaslan vd., 2009 tarafından yapılan çalışmaya istinaden, bu çalışmada kullanılan veri kümesi ile iki farklı destek vektör deneyi yapılmıştır. İlk deneyde Linear Kernel çekirdeği kullanılmış ve elde edilen doğru sınıflandırma oranı için 0,88 kappa değeri 0,47 ve f-ölçütü değeri 0,87 olmuştur. İkinci deneyde Radial Basis Function Kernel çekirdeği kullanılarak elde edilen doğru sınıflandırma oranı 0,85 kappa değeri 0 ve f-ölçütü değeri 0,79 dur. Bu sonuçlardan ikinci deneyde Radial Basis Function kernelini kullanan SMO algoritmasının, ifade gücü yüksek algoritmaların yaşadığı aşırı uyum (overfitting) sorunu ile karşılaştığı söylenebilir. Bunun yanında başarı oranı olarak doğrusal olan bir veri kümesinin analizinde en yüksek sonuçların doğrusal olarak düzenlenmiş veya güçlü bir algoritmanın en zayıf durumu ile alması beklenir. Yapılan deneylerde doğrusal çekirdeği kullanan SMO sonuçları da veri kümesinin doğrusal olduğunu göstermiştir.

4.5.4. J48 Karmaşıklık Deneyleri

Tablo 4.24 J48 Algoritması Karmaşıklık Deneyleri Sonuçları

Pruned	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
boost+unpruned	0,87	0,44	0,86	0,87	0,86	0,35
pruned	0,89	0,51	0,88	0,89	0,88	0,30

Bir karar ağacı algoritması olan J48, sınıflandırma açısından yüksek kapasiteye sahip doğrusal olmayan bir algoritmadır. Karar ağaçlarında budama yapmadan

algoritmanın sahip olduğu parametreleri değiştirilerek yapılacak güçlendirme işlemleri algoritmanın eğitim kümesine aşırı uyum göstererek ezberleme yapılmasına neden olacaktır. Yapılacak budama işlemleri ile de algoritmanın ifade gücünü düşürerek genelleme gücü arttırılabilir.

Kılıçaslan vd., (2009) yaptığı çalışma incelendiğinde veri kümesi üzerinde uygulanan j48 algoritması kullanarak, farklı iki parametre kullanarak iki deney gerçekleştirilmiştir. İlk deneyde j48 algoritmasında boosting (güçlendirme) ve unpruned (budamasız) özelliği uygulanarak doğru sınıflandırma oranı olarak 0,77 kappa değeri olarak 0,50 ve f-ölçütü olarak 0,67 sonuçları elde edilmiştir. İkinci deneyde j48 algoritmasında pruned (budama) özelliği uygulanarak doğru sınıflandırma oranı olarak 0,81 kappa değeri olarak 0,57 ve f-ölçütü olarak 0,71 sonuçları elde edilmiştir. Karar ağaçlarının boosting (güçlendirilme) özelliğinin kullanılması ve budama yapılmaması algoritmanın ifade gücünü arttırır bu da gürültülü verilerin yarattığı aşırı uyum problemini ortaya çıkarır. Bu durum ilk deneyde j48 algoritmasının aldığı değerlerin düşük çıkmasını açıklar.

Tezde kullanılan veri kümesi ile iki farklı j48 algoritması deneyi yapılmıştır. İlk deneyde j48 algoritmasında boosting (güçlendirme) ve unpruned (budamasız) özelliği uygulanarak elde edilen doğru sınıflandırma oranı için 0,8 kappa değeri 0,44 ve f-ölçütü değeri 0,86 olmuştur. İkinci deneyde pruned (budama) kullanılarak elde edilen doğru sınıflandırma oranı 0,89 kappa değeri 51 ve f-ölçütü değeri 0,88 dur. Bunun yanında başarı oranı olarak doğrusal olan bir veri kümesinin analizinde en yüksek sonuçların doğrusal olarak düzenlenmiş veya güçlü bir algoritmanın en zayıf durumu ile alması beklenir. Yapılan deneylerde budama gerçekleştirilen deneyden elde edilen sonuçların daha yüksek olması veri kümesinin doğrusal olduğunu göstermiştir.

4.5.5. Naïve Bayes Karmaşıklık Deneyleri

Tablo 4.25 Naïve Bayes Algoritması Karmaşıklık Deneyleri Sonuçları

Use kernel estimator	Correctly Classified	Kappa	Precision	Recall	F-Measure	RMSE
Normal	0,87	0,50	0,87	0,87	0,87	0,30
Kernel Estimator	0,87	0,50	0,87	0,87	0,87	0,30

Mannin vd.'ne göre Naïve Bayes, normal dağılım varsayımına sahip doğrusal bir sınıflandırıcıdır. John ve Langley'e göre Naïve Bayes çekirdek yoğunluk kestiricisi (kernel density estimator) kullanıldığında doğrusal olmayan bir sınıflandırıcıya dönüşebilmektedir.

Kılıçaslan vd., 2009 yaptıkları çalışmada uyguladıkları Naïve Bayes deneylerinde normal dağılım varsayımını çekirdek kestirimi ile değiştirmişlerdir. Bu f-ölçütü ve doğruluk değerlerinde az da olsa bir yükselişi beraberinde getirmiştir. Ancak tez çalışmasında kullanılan veri kümesinde üzerinde uygulanan bu iki deney sonucunda her hangi bir farklılık görülmemiştir.

4.5.6. Karmaşıklık Deneyleri Sonuçları

Bir önceki bölümde yapılan doygunluk deneylerinde elde edilen sonuçları yorumladığımızda oluşturduğumuz veri kümesinin doygun olduğu sonucuna varmıştık. Buna istinaden deneylerini yaptığımız Voted Perceptron algoritmasın sonuçlarında da en yüksek değerleri bulmayı bekleriz ki elde edilen 0,88 f-ölçütü değeri diğer tüm algoritmalar arasında en yüksek değerdir ve bu durum mevcut veri kümemizin lineer olduğunun bir ispatı olarak sunulabilir. Bunun yanında elde edilen 0,88 lik f-ölçütü değeri modelin tanı için yeterli olduğunu gösteren bir sonuçtur. Ayrıca Voted Perceptron algoritmasını ilgili deney için sahip olduğu kappa değerinin 0,46 olduğu görülmektedir. Bu değer yapılan deneyler arasında elde edilen en yüksek kappa değerleri arasında olmasıyla birlikte algoritmanın elde ettiği başarının şans eseri olmadığını bize göstermektedir.

BÖLÜM 5

SONUÇLAR

Bu çalışmada makine öğrenmesi algoritmaları kullanılarak eğitim alanlı bir veri kümesi üzerinde doygunluk ve karmaşıklık analizi gerçekleştirilmiştir.

Eğitim alanlı veri kümesi oluşturulması için Edirne merkezinde bulunan tüm ilköğretim okullarının 4.sınıfında okuyan öğrencilerin akademik başarıları ve kişisel bilgileri bir saha çalışması ile toplanmıştır. Optik formlarla toplanan veriler Microsoft Visual Studio 2010 C# kullanılarak geliştirilen bir program aracılığı ile Access veri tabanına kaydedilip saklanmıştır. Veritabanına kaydedilen veriler incelenerek anlamsız ve eksik bulunanlar veri kümesinden çıkarılmıştır. Bu temizleme işleminde Visual Studio 2010 C# kullanılarak geliştirilen program ve Weka programının Visualize ekranı kullanılmıştır.

Temizlik sonrası 570 örneğe sahip bir kaynak veri kümesi elde edilmiştir. Kaynak veri kümesi içinden tekrarsız seçim yapılarak 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 550 ve 570 örneğe sahip yeni veri kümeleri oluşturulmuş ve arff dosya formatında kaydedilmiştir.

Çalışmada, classify value (sontest) özniteliğinin sahip olduğu sınıf sayısının algoritma başarısında etkili olabileceği düşüncesiyle sahip olabileceği farklı sayıdaki sınıflar için dönüştürme işlemleri gerçekleştirilmiştir. Sontest özniteliğinde yapılan dönüştürme işlemi ile 21 sınıflı, 5 sınıflı ve 2 sınıflı veri kümeleri elde edilmiştir.

21 sınıflı sontest özniteliğine sahip veri kümesi 12 veri kümesine ayrılmış ve herbir veri kümesi için Weka'nın sahip olduğu sınıflandırma algoritmaları çalıştırılmıştır. Elde edilen sonuçlarda tüm veri kümeleri için kappa değerinin çalışmada esas alınan 0,4 değerinin altında olduğu tespit edilmiştir. Bu durum kullanılan veri kümelerinden elde edilen başarının rastlantısal olduğunu ve veri kümelerinin dengesiz olduğunu

göstermiştir. F-measure değerinin de 0,5 değerinin altında olması bu modelin tanı için yeterli olmadığını göstermiştir.

Öznitelik dönüştürme ile elde edilmiş 5 sınıflı sontest özniteliğine sahip veri kümesinden elde edilen 12 veri kümesinde de sınıflandırma algoritmaları çalıştırılmıştır. Bu veri kümesi için de kappa değerinin 0,4'ün altında olduğu görülmüş ve kullanılan veri kümelerinden elde edilen başarının rastlantısal olduğuna ve veri kümelerinin dengesiz olduğuna karar verilmiştir. F-measure değerinin 0,5 değerinin yakınında olması tek başına yeterli olmayacağından sınıf sayısının azaltılmasına karar verilmiştir.

Öznitelik dönüştürme işlemi ile 2 sınıflı sontest özniteliğine sahip veri kümesinden elde edilen 50,100, 150, … 550 ve 570 örneğe sahip toplam 12 alt veri kümesinde Weka 3.7. programının classify sekmesinde bulunan LWL, J48, JRIP, Part, LMT, Baggıng, Random Forest, IBK, MultiLayer Perceptron, Voted Perceptron, SMO, Naïve Bayes algoritmaları çalıştırılmıştır. Bu algoritmalara ait mutluluk grafikleri ve deney sonuçları incelendiğinde sadece Naïve Bayes ve Voted Perceptron algoritmalarında verinin örnekleme yoğunluğunun doygunluk seviyesine ulaşmış olduğu sonucu çıkarılabilmektedir.

Navie Bayes algoritmasının mutluluk grafiğine bakıldığında doygunluk başlangıç noktasının daha 100 örnekli ikinci veri kümesinde başladığı söylenebilir. Buradaki doğru sınıflandırma oranı 0,88 kappa değeri 0,47 ve f-ölçütü değeri 0,88 dir. Daha sonra 200, 350, 400, 450 örnekli veri kümelerine ait sonuçlarda f-ölçütü çizgisi dalgalı bir görünüm almıştır. Bu alt veri kümeleri için kirlilikten söz edilebilir.

Voted Perceptron algoritmasının mutluluk grafiğine bakıldığında f-ölçütünün 500 örneğe sahip veri kümesine kadar yükseldiğinin burada yakaladığı 0,89 değerini 570 örnekli veri kümesine kadar koruduğu ve bu noktada azalma eğilimine girdiği görülmektedir. Voted perceptron algoritmasının doygunluk başlangıç noktasının 500 örnekli veri kümesi olduğu söylenebilir bu noktada aldığı doğru sınıflandırma oranı 0,90 kappa değeri 0,52 ve f-ölçütü değeri 0,89 dur.

Yapılan tüm deneyler göstermiştir ki Naïve Bayes ve Voted Perceptron algoritmalarının tutarlı bir şekilde mutluluk çizgisi sunması kullanılan veri kümesinin aslında doygun olduğunu göstermektedir. Ancak diğer algoritmaların muhtemelen bir özelliğinden dolayı veya bu algoritmaların bu veri kümesindeki veriler için aşırı güçlü

olmasından dolayı kirliliğe çok fazla uyum göstererek mutluluk çizgisi elde edilemediği düşünülmektedir.

Veri kümelerinin karmaşıklığını belirlemek üzere literatürde karmaşıklık analizinde kullanılan IBK, SMO, Voted Perceptron, J48 ve Naïve Bayes algoritmaları 2 sınıflı sontest özniteliğine sahip veri kümesine uygulanmıştır.

Karmaşıklık analizinde kullanılan Voted Perceptron algoritmasının sonuçlarının yine en yüksek değerleri verdiği (0,88 f-ölçütü değeri) görülmüştür. Bu durum veri kümesinin lineer olduğunu göstermektedir. Elde edilen 0,88 lik f-ölçütü değeri modelin tanı için yeterli olduğunu gösteren bir sonuçtur. Voted Perceptron algoritmasının ilgili deney için sahip olduğu kappa değerinin 0,46 olduğu görülmektedir. Bu değer yapılan deneyler arasında elde edilen en yüksek kappa değerleri arasında olmasıyla birlikte algoritmanın elde ettiği başarının şans eseri olmadığını göstermektedir.

Verinin lineer olduğu durumlarda, başarılı sonuçlar veren bir algoritma olan Voted perceptron algoritması ile yapılan karmaşıklık deneylerinde farklı üs değerleri için algoritmanın lineerliği değiştirilmiştir. Üs değeri arttıkça doğru sınıflandırma oranının düştüğünün görülmesi kullanılan veri kümesinin lineer olduğunun göstergesi olmuştur.

IBK yakın komşu algoritması doğrusal olmayan bir algoritmadır ve sınıflandırma açısından yüksek kapasiteye sahiptir. IBK algoritmasında belirlenen k değerinin artması algoritmanın ifade gücünü arttırmaktadır. Ancak doğrusal olan veri kümelerinde deneyler yapılırken k değeri doğru seçilmediği takdirde aşırı uyum denilen durum gerçekleşir. IBK algoritması ile yapılan karmaşıklık deneylerinde k değerinin 15 ve 20 olarak kullanıldığı deneylerde f-ölçütünün 0,84 ve kappa değerinin 0,26 - 0,25 olması modelin eğitim kümesine aşırı uyum gösterdiğini yani overfitting olduğunu göstermektedir.

Destek karar makineleri (SVM) ile yapılan karmaşıklık analizinde veri kümesini farklı çekirdeklerle sınıflandırmak amacıyla SVM kullanılmıştır. Analiz sırasında çekirdek değiştirilerek güç arttırma işlemi ile yapılmıştır. Algoritmanın ifade gücünün artması ile SMO algoritması aşırı uyum (overfitting) sorunu ile karşılaşmıştır. Deneylerde doğrusal çekirdeği kullanan SMO sonuçlarının daha yüksek olması da veri kümesinin doğrusal olduğunu göstermiştir.

Sınıflandırma açısından yüksek kapasiteye sahip doğrusal olmayan bir algoritma olan J48 algoritmasında boosting (güçlendirme) ve unpruned (budamasız) özelliği uygulanarak elde edilen doğru sınıflandırma başarısı pruned (budama) kullanılarak elde

sınıflandırma başarısından daha düşük olmuştur. Yapılan deneylerde budama gerçekleştirilen deneyden elde edilen sonuçların daha yüksek olması veri kümesinin doğrusal olduğunu göstermiştir.

Naïve Bayes algoritması ile yapılan iki deneyde, normal dağılım varsayımını çekirdek kestirimi ile değiştirilmiştir. Veri kümesinde uygulanan bu iki deneyde sonuçlar arasında her hangi bir farklılık görülmemiştir.

KAYNAKLAR

[1] John Wiley & Sons, Inc.,***Big Data Analytics For Dummies***, ISBN 978-1-118-60704-6, 6, 2013

[2] Kahramanlı, H., ***Hibrit Bulanık Sinir Ağını Kullanarak Bir Sınıflandırma Ve Kural Çıkarma Sisteminin Geliştirilmesi***, Doktora Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, 2008

[3] Akın, Y., K., ***Veri Madenciliğinde Kümeleme Algoritmaları Ve Kümeleme Analizi***, Doktora Tezi, Marmara Üniversitesi Sosyal Bilimler Enstitüsü Ekonometri Anabilim Dalı, 2008

[4] Koçer, B., ***Transfer Öğrenmede Yeni Yaklaşımlar,*** Doktora Tezi ,Selçuk Üniversitesi Fen Bölümleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, 2012

[5] Kuzey, C., ***Veri Madenciliğinde Destek Vektör Makinaları ve Karar Ağaçları Yöntemlerini Kullanarak Bilgi Çalışanlarının Kurum Performansı Üzerine Etkisinin Ölçülmesi ve Bir Uygulama***, Doktora Tezi, İstanbul Üniversitesi Sosyal Bilimler Enstitüsü Sayısal Yöntemler Bilim Dalı, 2012

[6] Dener, M., Dörterler, M., Orman, A., "***Açık kaynak kodlu veri madenciliği programları: WEKA'da örnek uygulama***", Akademik Bilisim'09 - XI. Akademik Bilişim Konferansı, Şanlıurfa, 787-796 (2009)

[7] Yardımcı, T., ***Makine Öğrenmesi Teknikleri İle Rss Besleme Yönetimi,*** Ankara: Yüksek Lisans Tezi, Bilgisayar Eğitimi, Gazi Üniversitesi, Bilişim Enstitüsü, 2011

[8] Güner, N., Çomak, E., ***Mühendislik Öğrencilerinin Matematik I Derslerindeki Başarısının Destek Vektör Makineleri Kullanılarak Tahmin Edilmesi*** , Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi Cilt 17, Sayı 2, Sayfa 87-96, 2011

[9] Ünsal, Ö.,, ***Mesleki Alan Seçimlerinin Makine Öğrenmesi Algoritması kullanılarak Belirlenmesi***, Yüksek Lisans Tezi, Gazi Üniversitesi Bilişim Enstitüsü, 2011

[10] Dalyan, T., "***Makine öğrenmesinde 1R algoritması ve ikinci kuralın (2R) oluşturulması***", Yüksek Lisans Tezi, Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Kocaeli, 14, 17 (2006)

[11] Çelik, M., ***Veri Madenciliğinde Kullanılan Sınıflandırma Yöntemleri ve Bir Uygulama***, Yüksek Lisans Tezi, İstanbul Üniversitesi Sosyal Bilimler Enstitüsü, 2009

[12] Uzun, E., ***İnternet Tabanlı Bilgi Erişimi Destekli Bir Otomatik Öğrenme Sistemi***, Doktora Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü, 2007

[13] Güngör, İ., ***Yapay Sinir Ağları Yöntemi İle Tahmin (İmkb'de Bir Uygulama),*** Yüksek Lisans Tezi, Süleyman Demirel Üniversitesi Sosyal Bilimler Enstitüsü, 2003

[14] Aydın, F., ***Kalp Ritim Bozukluğu Olan Hastaların Tedavi Süreçlerini Desteklemek Amaçlı Makine Öğrenmesine Dayalı Bir Sistemin Geliştirilmesi,*** Yüksek Lisans Tezi, Trakya Üniversitesi Fen Bilimleri Enstitüsü, 2011

[15] Velickov, S. and Solomatine, D***., Predictive Data Mining: Practical Examples***, Artificial Intelligence in Civil Engineering, Germany, s.4, 2000
[16] Akpınar, H. ***Veri Tabanlarında Bilgi Kesfi ve Veri Madenciliği***, İ.Ü. İşletme Fakültesi Dergisi, c:29,S:1/Nisan 2000, s 5-6, 2000
[17] Terzi Ö., Küçüksille E. U., Ergin G., İlker A., ***Veri Madenciliği Süreci Kullanılarak Güneş Işınımı Tahmini***, SDU International Technologic Science,2011, 3, 29-37
[18] Aylık, Y.,Z., Özdemir, A., Yavuz, U., ***Lise Türü ve Lise Mezuniyet Başarısının, kazanılan Fakülte ile ilişkisinin Veri Madenciliği Tekniği ile Analizi***, Sosyal Bilimler Enstitüsü Dergisi, Cilt 10, no.2,p.441-454, 2007
[19] http://www.tameroz.com/Naïve Bayes-kullanarak-datamining.
[20] Altındağ,O., ***Kanser Sınıflandırmada MikroRNA Ve mRNA Anlatım Bilgilerinin Entegrasyonu*** , Yüksek Lisans Tezi , Başkent Üniversitesi Fen Bilimleri Enstitüsü, 2013
[21] Hamzaçebi C., Bayramoğlu F., Demirci E., "***Altın Fiyatının Ve İşlem Hacminin Tahmini***", Yöneylem Araştırması ve Endüstri Mühendisliği 30. Ulusal Kongresi, İstanbul, Türkıye, 2010. (http://slideplayer.biz.tr/slide/1932327/)
[22] Nöral sistemler Ders Notu: bm.bilecik.edu.tr/Dosya/Icerik/107/DosyaEki/nsg_ders_notu.pdf, Erişme tarihi: 18-06-2015
[23] Tanrıkulu,H., ***Saldırı Tespit Sistemlerinde Yapay Sinir Ağlarının Kullanılması,*** Yüksek Lisans Tezi, Ankara Üniversitesi Fen Bilimleri Enstitüsü, 2009
[24] Özhan,E., ***Yapay Zeka Yöntemleri İle Dünya Depremlerinin Modellenebilirliği***, ISITES2014 Karabuk – TURKEY,2014
[25] Çalışkan, S.K., Soğukpınar,İ., ***KxKnn: K-Means Ve K En Yakın Komşu Yöntemleri İle Ağlarda Nüfuz Tespiti.*** 2.Ağ ve Bilgi Güvenliği Ulusal Sempozyumu, 2008
[26] http://www.csharpnedir.com/articles/read/?id=811, Erişim Tarihi: 19.06.2015
[27] http://www.mademir.com/2010/12/k-nn-algoritmas.html, Erişim Tarihi: 19.06.2015
[28] Gebeyehu,S., Rao,V.,S., ***A Two Step Data Mining Approach for Amharic Text Classification*** ,American Journal of Engineering Research (AJER) e-ISSN : 2320-0847 p-ISSN : 2320-0936 Volume-03, Issue-04, pp-251-259, 2014
[29] Korkem,E***., Mikroarray Gen Ekspresyon Veri Setlerinde Random Forest Ve Naïve Bayes Sınıflama Yöntemleri Yaklaşımı***, Yüksek Lisans Tezi, Hacettepe Üniversitesi,Sağlık Bilimleri Enstitüsü, 2013
[30] Cingiz,M.,Ö., Albayrak,A., Amasyalı, M.F.,***21. IEEE Sinyal İşleme ve İletişim Uygulamaları (SİU2013) Kurultayı*** Uluslararası Kıbrıs Üniversitesi, 2013
[31] Kökçü,B.,N, Köse,R.,D., Bulut,F., Amasyalı,M.,F., ***Kolektif Öğrenme Algoritmalarıyla Çocuklarda Obezite Hastalığına Yakalanma Olasılıklarının Hesaplanması***, Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu, 09-10 Ekim 2014
[32] Uzun,Y., Tezel,G.,***Rule Learnıng Wıth Machıne Learnıng Algorıthms And Artıfıcıal Neural Networks***, Journal Of Selçuk Unıversıty Natural And Applıed Scıence 2012 Vol.1 No.2 Onlıne Issn:2147-3781, 2012
[33] S.Vijayarani M.Divya, ***An Efficient Algorithm for Classification Rule Hiding***, International Journal of Computer Applications (0975 – 8887) Volume 33– No.3, November 2011

[34] Eibe F, Ian H. W., ***Generating Accurate Rule Sets Without Global Optimization,*** In: Fifteenth International Conference on Machine Learning, 144-151, 1998

[35] İçelen,N.,***Veri Madenciliği Yöntemi İle Divriği Nuri Demirağ Meslek Yüksekokulu Öğrencilerinin Temel Bilgisayar Dersine Ait Başarı Analizi Uygulaması***, MBD 2012, 1(1): 18 – 37, 2012

[36] Yuksek,M.,C., Barshan,B., ***Minyatur Eylemsizlik Duyucuları ve Manyetometre Sinyallerinin İşlenmesiyle İnsan Aktivitelerinin Sınıflandırılması***,2011 IEEE 19th Signal Processing and Communications Applications Conference (SIU 2011)

[37] Daş,B.,Türkoüğlu,İ., ***DNA Dizilimlerinin Sınıflandırılmasında Karar Ağacı Algoritmalarının Karşılaştırılması,*** Eleco 2014 Elektrik – Elektronik – Bilgisayar ve Biyomedikal Mühendisliği Sempozyumu, 27 – 29 Kasım 2014, Bursa,2014

[38] Okur,B.,Ç.,***İngiliz Alfabesi Kullanılarak Yazılmış Türkçe Metinlerin Türk Alfabesine Göre Yeniden Oluşturulması***, Yüksek Lisans Tezi, Gebze Yüksek Teknoloji Enstitüsü Mühendislik Ve Fen Bilimleri Enstitüsü, 2013

[39] Akçetin,E.,Çelik,U., ***İstenmeyen Elektronik Posta (Spam) Tespitinde Karar Ağacı Algoritmalarının Performans Kıyaslaması***,İnternet Uygulamaları ve Yönetimi, Süreli Dergi, IUYD'2014 / 5(2), 2014

[40] Kumar, P., Wasan,S.,K., ***Analysis of cancer datasets using C Analysis of cancer datasets using Classification Algorithms***,,IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.6, June 2010

[41] Kaladhar,D.,, Chandana,B., ***Data mining, inference and prediction of Cancer datasets using learning algorithms,***International Journal of Science and Advanced Technology (ISSN 2221-8386) Volume 1 No 3, 2011

[42] Akman, M., Genç,Y., Ankaralı H, ***Random Forests Yöntemi ve Sağlık Alanında Bir Uygulama***, Turkiye Klinikleri J Biostat 2011;3(1), p:36-48, 2011

[43] Yılmaz,H., ***Random Forests Yönteminde Kayıp Veri Probleminin İncelenmesi Ve Sağlık Alanında Bir Uygulama***, Yüksek Lisans Tezi,Eskişehir Osmangazi Üniversitesi, Sağlık Bilimleri Enstitüsü, 2014

[44] Yıldırım, S., ***Türkçe Derlemlerdeki Artgönderimlerin Tümdengelimli ve Tümevarımlı Yöntemlerle Çözümlenmesi***, Doktora Tezi, Trakya Üniversitesi Fen Bilimleri Enstitüsü, 2008

[45] Güner, E. S., ***Türkçe İçin Derlem Tabanlı, Bir Anafor Çözümleme Çalışması***, Yüksek Lisans Tezi, Trakya Üniversitesi Trakya Üniversitesi Fen Bilimleri Enstitüsü, 2008

[46] Platt, J.C., ***Sequential Minimal Optimization: A Fast Algorithm for Training Support Vektor Machines,*** Microsoft Research, Technical Report MSR –TR-98-14, 1998.

[47] Göker, H., ***Üniversite Giriş Sınavında Öğrencilerin Başarılarının Veri Madenciliği Yöntemleri İle Tahmin Edilmesi,*** Yüksek Lisans Tezi, Gazi Üniversitesi Bilişim Enstitüsü, 2012

[48] Coşkun,C., Baykal, A., ***Veri Madenciliğinde Sınıflandırma Algoritmalarının Bir Örnek Üzerinde Karşılaştırılması***, Akademik Bilişim'11 - XIII. Akademik Bilişim Konferansı Bildirileri 2 - 4 Şubat 2011 İnönü Üniversitesi, Malatya, 2011

[49] Nizam, H., Akın, S.,S., ***Sosyal Medyada Makine Öğrenmesi ile Duygu Analizinde Dengeli ve Dengesiz Veri Setlerinin Performanslarının Karşılaştırılması***, XIX Türkiye İnternet Konferansı, 2014

[50] Kılıçaslan Y., Güner E. S., Yıldırım S., ***Learning-based pronoun resolution for Turkish with a comparative evaluation***, Computer Speech & Language Volume 23, Issue 3, July 2009, Pages 311-331, 2009.

[51] Ardıl,E., ***Esnek Hesaplama Yaklaşımı İle Yazılım Hata Kestirimi***, Yüksek Lisans Tezi,Trakya Üniversitesi Fen Bilimleri Enstitüsü, 2009

[52] Uçar, E., ***Ortaöğretime Geçiş Sistemi (Oges) Yerleştirme Puanlarının Uzman Sistemler İle Tahmini***, Doktora Tezi, Karabük Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, 2013

[53] Haciefendioğlu, Ş., ***Makine Öğrenmesi Yöntemleri İle Glokom Hastalığının Teşhisi***, Yüksek Lisans Tezi, Selçuk Üniversitesi, Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Sistemleri Anabilim Dalı, 2012

[54] http://scikit-learn.org/stable/auto_examples/plot_underfitting_overfitting.html Erişim zamanı: 20.10.2015

[55] Özarslan S., Barışçı N., ***Öğrenci Performansının Veri Madenciliği İle Belirlenmesi***, Isıtes2014 Karabük – Türkiye Sayfa 1958-1965, 2014

[56] Alan M. A., ***Veri Madenciliği ve Lisansüstü Öğrenci Verileri Üzerine Bir Uygulama***, Dumlupınar Üniversitesi Sosyal Bilimler Dergisi, Sayı 33, Sayfa.165-174, 2012

[57] Ekren G., ***Bir Veri Madenciliği Tekniği İle Öğrenci Segmentasyonu***, 21. Ulusal Eğitim Bilimleri Kongresi, 2012

[58] Kartal, E., ***Sınıflandırmaya Dayalı Makine Öğrenmesi Teknikleri Ve Kardiyolojik Risk Değerlendirmesine İlişkin Bir Uygulama***, Doktora Tezi, İstanbul Üniversitesi Enformatik Anabilim Dalı, 2015

[59] Çatak,F.,Ö., ***Bulut Bilişim Sistemlerinde Eşle/İndirge Yöntemi Uygulanarak Veri Madenciliği Yazılım Çatısının Geliştirilmesi***, Doktora Tez, İstanbul Üniversitesi Fen Bilimleri Enstitüsü Enformatik Anabilim Dalı, 2014

[60] Emir, Ş., ***Yapay Sinir Ağları ve Destek Vektör Makineleri Yöntemlerinin Sınıflandırma Performanslarının Karşılaştırılması: Borsa Endeks Yönünün Tahmini Üzerine Bir Uygulama***, Doktora Tezi, İstanbul Üniversitesi Sosyal Bilimler Enstitüsü Sayısal Yöntemler Bilim Dalı, 2013

[61] Aydilek, İ., B., ***Veri Kümelerindeki Eksik Değerlerin Yeni Yaklaşımlar Kullanılarak Hesaplanması,*** Doktora Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, 2013

[62] Aslan, B., G, ***E-Öğrenme İçin Öğrenci Modellemesine Yönelik Bir Orta Katman Yapınının Gerçeklestirilmesi***, Doktora Tezi, Ege Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı Bilim Dalı, 2010

[63] Köktürk, F., ***K-En Yakın Komşuluk, Yapay Sinir Ağları Ve Karar Ağaçları Yöntemlerinin Sınıflandırma Başarılarının Karşılaştırılması***, Doktora Tezi, Bülent Ecevit Üniversitesi Sağlık Bilimleri Enstitüsü Biyoistatistik Anabilim Dalı, 2012

[64] https://tr.wikipedia.org/wiki/Weka, Erişim Tarihi:23.06.2015

[65] http://bm.bilecik.edu.tr/Dosya/Icerik/107/DosyaEki/nsg_ders_notu.pdf Erişim tarihi: 23.06.2015

[66] Şişaneci, İ., vd., ***Performansa Dayalı Ek Ödeme Sisteminde Veri Madenciliği İle Suistimal Tespit***, Computer Science and Engineering Research Journal, 2010.

[67] Şaylan, Ç., A., ***Böbrek Nakli Geçirmiş Hastalarda Akıllı Yöntem Tabanlı Yeni Öznitelik Seçme Algoritması Geliştirilmesi,*** Kadir Has Üniversitesi Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2013.

[68] Petra Perner (ed.) ***Machine Learning and Data Mining in Pattern Recognition***, 6th International Conference, MLDM 2009, Leipzig, Germany, 754-755, July 2009, Proceedings, ISSN:0302-7743

ÖZGEÇMİŞ

1979 yılında Edirne'de doğmuştur. İlk ve orta öğrenimini tamamladıktan sonra Trakya Üniversitesi Bilgisayar Programcılığı bölümünü bitirmiştir. 2002 yılında Edirne' de bulunan özel bir lisede Bilgisayar Öğretmeni olarak meslek hayatına başlamıştır. Yeniliklere açık olan ve kendisine bu anlamda yatırım yapan Tolga Demirhan yaşam boyu öğrenmeyi ilke edinmiştir. Farklı bakış açıları kazanmaları, problem çözme ve araştırma becerilerini geliştirmeleri için öğrencilerini bilimsel yarışmalara katılmaları için teşvik etmiştir. Bu bağlamda yetiştirdiği öğrenciler ile katıldığı birçok ulusal ve uluslararası bilimsel yarışmalarda ödüller kazanmıştır. Çalışmaları yanında eğitim hayatına da devam ederek Anadolu Üniversitesi İşletme bölümünü bitirmiş ve Trakya Üniversitesi Bilgisayar Mühendisliği bölümünde Yüksek Lisans yapmıştır. 2010 yılı başında Trakya Üniversitesi Tunca Meslek Yüksekokulu'na öğretim görevlisi olarak atanmıştır. 2014 yılında almış olduğu bir incelemeli patent ile bir faydalı model tesciline sahiptir. Ayrıca yazarları arasında olduğu iki kitap bugün Trakya Üniversitesi'nde BIL117 Temel Bilgi Teknolojileri 1 ve BIL118 Temel Bilgi Teknolojileri 2 derslerinde kaynak kitap olarak okutulmaktadır. Evli ve bir çocuk babasıdır.

TEZ ÖĞRENCİSİNE AİT TEZ İLE İLGİLİ BİLİMSEL FAALİYETLER

Analysing The Performances Of 4th Year Primary School Students In Edirne In Science And Technology Course, "Let's Solve The Riddle: Our Body" Chapter Through Data Mining, Tolga DEMİRHAN, Özlem UÇAR, 6.International Education, Research & Development Conferans, 4-8 September 2015.

A Problem Based Learning Application İn Primary Level Of Deaf Education, Tolga DEMİRHAN, Özlem UÇAR, Erdem UÇAR, 8th International Computer & Instructional Technologies Symposium, 18- 20 September 2014.

EK-A İl Milli Eğitim Müdürlüğü İzin Belgesi

T.C.
EDİRNE VALİLİĞİ
İl Millî Eğitim Müdürlüğü

Sayı : 56569733/44/3421275 18/08/2014
Konu : Anket Test ve Probleme Dayalı Öğrenme Uygulaması

TRAKYA ÜNİVERSİTESİ REKTÖRLÜĞÜNE
(Öğrenci İşleri Daire Başkanlığı)
EDİRNE

İlgi : 13/08/2014 tarihli ve 16031 sayılı yazınız

Üniversiteniz Fen Bilimleri Enstitüsü Bilgisyar Mühendisliği Anabilim Dalı Doktora Programı öğrencisi Tolga DEMİRHAN'ın, İlimiz Merkez ilkokullarında uygulamak istediği "Makine Öğrenmesi Metodları Kullanılarak, Probleme Dayalı Öğrenme Uygulamasının Başarı Analizi Yapılması Üzerine Bir Uygulama" konulu doktora tezi ile ilgili yapacağı çalışmalarda kullanacağı anket, test ve Probleme dayalı öğrenme uygulaması 2014-2015 eğitim öğretim yılı başladığında Anket Değerlendirme Komisyonunca değerlendirilecektir.

Bilgilerinizi ve gereğini rica ederim.

Dr. Ayhan ÖZKAN
Vali a.
Vali Yardımcısı

Güvenli Elektronik İmzalı
Aslı ile Aynıdır.
19/Ağustos/2014

Bu belge, 5070 sayılı Elektronik İmza Kanununun 5 inci maddesi gereğince güvenli elektronik imza ile imzalanmıştır
Evrak teyidi http://evraksorgu.meb.gov.tr adresinden 2cbf-1c58-38b2-bbcb-17d8 kodu ile yapılabilir.

http://edirne.meb.gov.tr/ Tel : (0284) 212 61 22 Yenilik ve Eğitim Teknolojileri Bölümü
bilgiislemegitek22@meb.gov.tr Faks : (0284) 212 61 26 Bilgi için: Saime GÖKKAYA (VHKİ) Dahili: 2303

FORM: 2

T.C.
MİLLÎ EĞİTİM BAKANLIĞI
Eğitimi Araştırma ve Geliştirme Dairesi Başkanlığı
ARAŞTIRMA DEĞERLENDİRME FORMU

ARAŞTIRMA SAHİBİNİN	
Adı Soyadı	Tolga DEMİRHAN
Kurumu / Üniversitesi	T.Ü. Fen Bilimleri Enstitüsü
Araştırma Yapılacak İl	Edirne Merkez
Araştırma Yapılacak Eğitim Kurumu ve Kademesi	Edirne Merkez İlkokul 4.Sınıf Öğrencileri
Araştırmanın Konusu	Makine Öğrenmesi Metodları Kullanılarak, Probleme Dayalı Öğrenme Uygulamasının Başarı Analizinin Yapılması Üzerine Bir Uygulama
Üniversite / Kurum Onayı	Var (X) / Yok ( )
Araştırma/Proje/Ödev /Tez önerisi	Tez Çalışması
Veri Toplama Araçları	Anket Formu (1 Sayfa) Test Formu (1 Sayfa) Probleme Dayalı Öğrenme Uygulaması (10 Sayfa)
Görüş İstenilecek Birim/Birimler	T.Ü. Fen Bilimleri Enstitüsü
KOMİSYON GÖRÜŞÜ	
Anketin uygulanması sırasında gönüllülük ilkesine uyulması esastır.	
Komisyon kararı	Oybirliği / Oyçokluğu ile alınmıştır.
Muhalif Üyenin Adı ve Soyadı:	Gerekçesi; .....

KOMİSYON

12/09/2014	12/09/2014	12/09/2014
Komisyon Başkanı	Üye	Üye
Fatih İNCE	Ayla DİKMEN	Ali ÇELİK
MEM Şube Müdürü	And. Öğretmen Lisesi Öğrt.	Emel Özgür S. Mes.veTek.And.Lis.Öğrt.