T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR ÖĞRENMESİNDE YENİ TÜMEVARIMSAL ALGORİTMALAR

DOKTORA TEZİ

Günay KARLI

Enstitü Anabilim Dalı	**:**	**ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ**
Enstitü Bilim Dalı	**:**	**ELEKTRONİK MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Doç. Dr. Ayhan ÖZDEMİR**

Ağustos 2010

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR ÖĞRENMESİNDE YENİ TÜMEVARIMSAL ALGORİTMALAR

DOKTORA TEZİ

Günay KARLI

Enstitü Anabilim Dalı : ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ

Bu tez 02/08/2010 tarihinde aşağıdaki jüri tarafından Oybirliği ile kabul edilmiştir.

Prof. Dr.
Etem KÖKLÜKAYA
Jüri Başkanı

Prof. Dr.
Ahmet Sertbaş
Üye

Doç. Dr.
Ayhan ÖZDEMİR
Üye

Doç. Dr.
Cabir VURAL
Üye

Yrd. Doç. Dr.
Tuğrul YANIK
Üye

TEŞEKKÜR

Öncelikler, bu tezin hazırlanmasında, doğrudan ve dolaylı olarak katkıda bulunan herkese teşekkür ediyorum.

Doktora çalışmalarımın başlamasından itibaren, rehberliğinden, tavsiyelerinden, eleştirilerinden ve cesaretlendirmelerinden dolayı, danışmanım Doç. Dr. Ayhan Özdemir Bey'e şükranlarımı ifade etmek istiyorum. Benim için kapısını her zaman açık tutan bir danışmanım olduğu için kendimi çok şanslı görüyorum.

Ayrıca, tez çalışmalarımız boyunda sağladıkları desteklerden dolayı, sayın Prof. Dr. Etem Köklükaya ve Doç. Dr. Cabir Vural Bey'e teşekkür etmek istiyorum.
Özellikle, eşsiz sabırlarından ve moral desteklerinden dolayı, eşim Elçin Karlı, çocuklarım Kerem Emre ve İrem'e teşekkür ediyorum.

İÇİNDEKİLER

BÖLÜM 3.

BÖLÜM 4.

SİMGELER VE KISALTMALAR LİSTESİ

$P(V_C)$:Veri setindeki her değerin sınıf bazındaki koşullu olasılıkları,

$|V_C|$: C sınıfında olan V değerlerinin sayısı,

$|V|$: Veri setindeki tüm V değerlerinin sayısı,

$T_{V,C}$: V değerinin C sınıfındaki olasılığının tümleyeni,

$E(V)$: Veri setindeki değerlerin entropileri,

$E(X)$: Veri setini oluşturan karakteristiklerin entropileri,

$M(V|C)$: Veri setindeki değerlerin sınıf bazındaki maliyetleri,

$ke_{v\text{-}c}$: Bir değerin belirli bir sınıftaki kurallaşma eğilimi,

DR_V : Bir V değerinin sınıflara dağılım oranı,

POW_A : Bir karakteristiğin sınıflandırma gücü,

$AVG()$: Ortalama fonksiyonu.

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Bilgisayar öğrenmesi, sınıflama, veri madenciliği, tümevarımsal öğrenme.

Bilgisayar teknolojisinin ilerlemesine paralel olarak, veri toplama kaynakları da oldukça gelişmiş ve yaygınlaşmıştır. Hemen her disiplin, kendi ürettiği veriler ile bereber dış çevreden elde ettiği verilerle, çeşitli amaçlar için büyük boyutlarda veritabanları oluşturmuştur. Bu verilerin içerisinde çok değerli ve gelecek için karar vermeyi sağlayacak bilgiler mevcuttur. Günümüzde bu veritabanlarından anlamlı ve yararlı bilgiyi ortaya çıkarabilmek önemli bir araştırma alanı haline gelmiştir.

Uzman sistemlerin en önemli görevlerinden biri, veritabanları içerisindeki yararlı, fakat saklı bilginin ortaya çıkartılmasıdır. Bilginin elde edilmesi, bilgi kazanımı olarak da isimlendirilir. Uzman sistemlerde, bilginin gösterilmesi, kullanılması ve kazanılması ile ilgili çözülmesi gereken önemli problemler bulunmaktadır. Bunlar içerisinde bilgi kazanımı problemi en kritik aşamayı oluşturmaktadır. Bu çalışmada, bilgiyi elde etmek amacı ile geliştirilen makine öğrenmesi algoritmaları (IREM ve ke-REM) sunulacaktır.

NEW ALGORITHMS IN INDUCTIVE MACHINE LEARNING

SUMMARY

Key words: Machine Learning, classification, data mining, inductive learning.

Parallel to the advancement in machine learning, data collecting sources were developed and became common. Almost all the disciplines formed a database with the data they produced along with those they collected from the surroundings. Among the data, there is valuable information that they may facilitate to give decisions about the future. It has been a significant research area to extract significant and useful knowledge from the databases.

One of the most significant tasks of expert systems is to explore useful but hidden information in databases. Acquisition of knowledge is also known as obtaining knowledge. There are vital problems that must be solved related to the illustration, usage and acquisition of information in expert systems. Acquisition of knowledge is the most crucial of them. Machine learning algorithms (IREM and ke-REM) developed to acquire information will be presented in this study.

BÖLÜM 1. GİRİŞ

Karmaşık üretim süreçleri genellikle değiştirilebilir kontrol parametrelerini içermektedir. Örneğin, işlenmemiş petrolü doğal gazdan ayırmak, petrol rafinesinin vazgeçilmez bir öngereksinimidir ve ayırma sürecini kontrol etmek hassas bir iştir. British Petroleum parametre ayarlamaları ile alakalı kural oluşturmak için bilgisayar öğrenmesini (machine learning) kullanmıştır. Böylece, uzman insanların bir günden daha fazla bir sürede yaptığı iş dakikalar seviyesine inmiştir [1]. Westinghouse, nükleer yakıt misketlerini üretme sürecinde problemler yaşamış ve süreci kontrol etmek için bilgisayar öğrenmesini kullanmıştır. Sonuç olarak, bu işlemin kendilerine 10 milyon dolar kazandırdığını rapor etmişlerdir [2]. Benzer şekilde, Tennessee baskı şirketi R.R. Donelly uygun olmayan parametre ayarlarının sebep olduğu nesneleri azaltmak için kullanılan bakır mardane (rotogravure) baskıyı kontrol etmede, nesnelerin sayısının yıllık 500'den 30'a indirmek için, aynı fikri uygulamıştır [3][4].

Benzer şekilde, Bell Atlantic haberleşme şirketinde, müşteri destek ve servis bölümüne, müşterilerinin telefonla alakalı şikayetleri ulaştığında, şirketin nasıl bir teknisyen tahsis etmesi gerektiğine karar vermesi bir problem haline gelmiştir. Bell Atlantic 1999 yılında, bu problemi çözmek için bilgisayar öğrenmesi ile karar mekanizması oluşturarak yılda 10 milyon dolardan daha fazla bir kazanç sağladığını rapor etmiştir [5].

Günümüzde, bankalardan kredi çekmek yaygın hale gelmiş operasyonlar arasındadır. Banka yetkilileri, yoğun talepler karşısında insan uzmanların taleplere cevap verme konusuda yetersiz kaldığını farketmiştir. Bu operasyonları yönetebilmek için bilgisayar öğrenmesi metodları geliştirilmiştir [6][7][8].

Eletronik dökümanların ulaşılabilirliğinin artması ve bir doküman okyanusu olan internet ortamının hızla yaygınlaşması ile birlikte, belgelerin otomatik olarak sınıflandırılması ve bilgi keşfi konusunda bir çok metod geliştirilmiştir. Çevrim içi haberler, günlükler, e-postlar, e-dökümanlar ve dijital kütüphanelerin uygun şekilde sınıflandırılmaları, bilgisayar öğrenmesi algoritmaları ile geliştirilen yöntemler sayesinde ilgililere kolaylıklar sağlanmaktadır [9][10][11].

Endüstrideki ugulamalarının yanında, bilgisayar öğrenmesinin kullanıldığı bir çok bilimsel uygulama da vardır. Biyolojide bilgisayar öğrenmesi yeni bir genomdaki binlerce geni tanımlamaya yardımcı olamada kullanılır [12][13][14]. Genetiğe bağlı bir takım hastalıkların teşhisinde de kullanılmaktadır [15][16][17]. Biyotıpta sadece ilaçların kimyasal özelliklerinin değil aynı zamanda üç boyutlu yapılarının analizi ile de ilaç aktiviteleri ile alakalı tahminde bulunmada kullanılır. Bu uygulama ilaç keşiflerini hızlandırır ve maliyetini de azaltır. Astronomide, görsel denetimle görülemeyecek kadar silik gökyüzü cisimlerinin otomatik kataloglama sistemini geliştirmede bilgisayar öğrenmesi kullanılmaktadır. Kimyada, manyetik rezonans spektradan bazı organik bileşenlerin yapılarını tahmin etme için kullanılmaktadır.
Yukarıda değinmeye çalışılan tüm bu uygulamalarda, bilgisayar öğrenmesi teknikleri dikkate değer performans dereceleri (beceri) kazanmıştır ki, bu beceri insan uzmanlar ile rekabet eder hatta geçer seviyededir [5] [18].

Bilgisayar öğrenmesi araştırmaları, bilgisayar alanındaki gelişmelere paralel olarak başlamış ve yürütülen çalışmaların sonucunda birçok metod ve algoritma geliştirilmiştir. Bu tezde geliştirilen keREM ve IREM algoritmalarının içinde bulunduğu mantıksal/sembolik bilgisayar öğrenme algoritmaları kategorisinde, böl ve fethet yaklaşımını kullanan karar ağaçları ile kapsama metodunu kullanan doğrudan kural üreten algoritmalar bulunmaktadır.

Karar ağacını kullanan ilk algoritma, Hunt [19] tarafından geliştirilen CLS (Concept Learning Sytem) algoritmasıdır. CLS serisi algoritmaların toplam sayısı dokuzdur. Bunlardan ilk sekizi sadece iki sınıflı örnek setleri ile çalışabilirken, CLS9 algoritmasında çok sınıflı örnekler çözülebilmektedir. Daha sonra Quinlan [20] tarafından ve bu metodu kullanan ID3 ailesi algoritmaları geliştirilmiştir. (ID3, ASSISTANT ve C4.5). ID3 algoritmasında, ayrıştırmaya rehberlik edecek bir bilgi

ölçüsü (entropi) kullanılmıştır. ID3 algoritması bilgi kazancını maksimize eden bir karakteristiği kök düğüm olarak seçer. Her düğüm için ilgili alt setin bilgi kazancı hesaplanarak bu düğüme göre alt setlere ayrıştırma işlemine, her düğümde tek sınıf kalıncaya kadar devam edilir. Daha sonra, ID3 endüktif öğrenme algoritmalarının bir serisi geliştirilmiştir. Bu seri ID3 ailesi algoritmaları olarak adlandırılır ID3-IV, GID3, ID4, ID5, ID5R ve ID5R-hat [21].

ID3 algoritmasının devamı olarak geliştirilen C4.5 algoritması, ID3 algoritmasının endüstriyel versiyonu olarak kabul edilmekte ve bugün birçok uzmanlık alanında yaygın olarak kullanılmaktadır [22]. ID3'ün gelişmiş bir şekli olan C4.5 algoritmasının en büyük özelliği karar ağacının gereksiz dallarını tespit ederek budamasıdır. Bu algoritmanın bir diğer özelliği ise eksik veri ve sayısal değerlerle de çalışabilmesidir.

Karar ağacı üreten bir diğer algoritma ise Breiman tarafından geliştirilmiştir: CART (Classification and Regression Trees) [23]. CART, her düğümündeki tüm karakteristikleri birer birer araştırır. Her bir karakteristik için en çok katkıyı sağlayan en iyi ayrıştırmayı bulur ve n adet aday içerisinden en iyi ayrıştırmayı seçer. Daha sonra karar ağacının kısa olmasını sağlamak için Crawford [24] tarafından CART algoritmasının uzantısı olan OC1 algoritması ile GDT-NR ve GDT-RS [25] algoritmaları geliştirilmiştir.

Rastogi ve Shim, ağaç budama yaklaşımına farklılık getirerek yeni bir karar ağaç sınıflandırıcısı olan PUBLIC algoritmasının geliştirdiler [26].

Karar ağaçlarından kurallar üretmek ve parçala-yönet öğrenme tekniklerini birleştiren PART algoritması, Markovitch and Rosenstein tarafından geliştirilmiştir. PART kısmi karar ağaçları ile kurallar çıkarır [27].

LMT (Logistic Model Trees) algoritması, mantıksal regrasyon ile ağaç yapısının birleştirildiği bir metodtur. Standart ağaç yapısı, yapraklarında regrasyon fonksiyonlarını barındırmaktadır [28].

Ayırma ölçütü olarak, bilgi kazancını kullanan diğer bir algoritma RepTree[29] algoritmasıdır. C4.5 algoritmasının farklı versiyonlarındandır ve hızlı çalışan karar ağaç yapısı üretir.

Küçük ve doğruluk oranı yüksek bir algoritma oluşturmayı hedefleyen RBDT (Rule-Based Decision Tree) [30] algoritması kurallarını sabit veya dinamik olarak değişen veri setlerinden oluşturmaktadır.

Karar ağaçlarının yanında, doğrudan kural üreten bir çok algoritma geliştirilmiştir. Bu alandaki ilk algoritma Michalski tarafından geliştirilen AQ[31] algoritmasıdır. Daha sonra AQ'nun bir çok versiyonları tasarlanarak AQ Ailesi algoritmaları üretilmiştir. AQ algoritmasının en önemli özelliği, tümevarımsal metodların kullandığı en temel yaklaşımlardan biri olan kapsama metodunu ilk kullanmasıdır. Daha sonraki yıllarda, kapsama yaklaşımını kullanan çok sayıda algoritma geliştirilmiştir: CN2[32], ITRULE, RISE, IREP, RIPPER[33], FOIL[34], Swap-1, RULES ailesi [35], ILA-2 [36], GDT-NR, GDT-RS [7] ve REX [37].

Bu algoritmalar arasındaki en önemli fark, veri setindeki örneklerin bilgi değerlerini dikkate alma noktasındadır. CN2, RULSE3 gibi algoritmalar bilgi değerleri ile ilgilenmezler. Bunun yanında RULSE4, ILA2, REX algoritmaları örneklerin bilgi değerlerini göz önünde bulundurarak daha yüksek doğruluk oranlarına ulaşmışlardır. Ayrıca, geliştirilen algoritmaların doğruluk oranlarını yükseltme adına, oluşturulan kuralların kalitesi [38][39], kuralarda çakışma[40] gibi konularda araştırmalar yapılmıştır ve devam etmektedir.

1.1. Tezin Amacı

Bu tezin amacı, bilgisayar öğrenmesi alanında, sınıflandırma işlemi gerçekleştiren yeni tümevarımsal algoritmalar geliştirmektir. Günümüzde, bir çok disiplin tarafından oluşturulmuş veri tabanlarında (veri seti, eğitim seti), yöneticilerin ve araştırmacıların kullanabileceği faydalı ama saklı bilgiler bulunmaktadır. Veri setlerindeki bu bilgilerin bilgisayar tarafından sağlanması, otomatik bilgi kazanımı olarak isimlendirilir.

Otomatik bilgi kazanımını sağlayan algoritmalar, üzerinde çalışılan veri setinden istifade ederek bir kural tabanı oluştururlar. Bir çok kayıt (veri, örnek) barındıran veri tabanlarındaki bazı kayıtlar, kural tabanın genelliği ve kapsayıcılığı açısından çok önemli olmakla birlikte, bazı kayıtlar önemsiz olarak nitelendirilir. Bundan dolayı, algoritmaların kural tabanını oluşturma süreçlerinde, önemli olarak nitelendirilen, bir başka deyişle, bilgi değeri yüksek kayıtların belirlenmesi, ve bu kayıtlar kullanılarak kural üretilmesi, bilgisayar öğrenmesi alanında çalışan araştırmacılar için önemli bir problem haline gelmiştir.

Bu çalışmada, sınıflandırma işlemini gerçekleştirmek için oluşturulacak kural tabanında, kullanılan eğitim setindeki verilerden bilgi değerleri yüksek olanlar ön plana çıkarılmıştır. Bu bağlamda, keREM ve IREM ismi verilen algoritmalarda bilgi değeri yüksek verilerin seçilmesi için maliyet ve kazanç formülleri oluşturulmuştur.

Maliyet fonksiyonu;

$$M(V|C) = \begin{cases} 0, & P(V_c) = 1 \\ -1, & P(V_c) = 0 \\ T_{V_{iC}} * E(V) + T_{V/C} * E(X), & 0 < P(V_c) < 1 \end{cases} \quad (1.1)$$

Geliştirilen maliyet fonksiyonunda, eğitim setindeki örnekleri oluşturan karakteristik-değer ikililerinin olasılık dağılımları ve entropileri kullanılarak, sınıf bazında bilgi değerleri hesaplaması gerçekleştirilmiştir.

Kazanç fonksiyonu;

$$ke_{V-C} = \begin{cases} 0, & P(V_c) = 0 \\ 1, & P(V_c) = 1 \\ AVG(\mathrm{P}(\mathrm{V_C}), \mathrm{DR_V}, POW_A), & 0 < P(V_c) < 1 \end{cases} \quad (1.2)$$

keREM algoritmasının kullandığı kazanç fonksiyonuna kurallaşma eğilimi ismi verilmiştir. Bu fonksiyonun temelini, karakteristik-değer ikililerinin olasılık dağılımları, değerlerin sınıf dağılım oranları ve karakteristiğin sınıflandırma gücü parametreleri oluşturmaktadır.

Ayrıca, keREM ve IREM algoritmalarının içinde yer aldığı mantıksal ve sembolik bilgisayar öğrenmesi kategorisindeki algoritmaların kural üretiminde, oluşturulan kuralların kalitesi kavramı önemli bir kriterdir. Yani, aday kuralın, eğitim setindeki kaç örneği kapsadığı, yüzde kaç doğrulukla tanıdığına ilişkin kriterler önem kazanmaktadır. keREM ve IREM'de, kural tasarımında kullanılan karakteristik-değer ikililerine ait bilgi değerleri maliyet ve kazanç fonksiyonları ile hesaplandığı ifade edilmişti. keREM ve IREM algoritmalarının kural oluşturma aşamasında bilgi değeri en yüksek karakteristik-değer ikililerini tercih eder. Bundan dolayı, oluşturulmuş kuralların kalite seviyesinin yüksek olduğu düşüncesi, gerçekleştirilen testlerle desteklenmektedir.

Geliştirilen maliyet ve kazanç fonksiyonlarının, bilgi değeri hesaplamasında daha hassas bir ölçüm sağladığı, keREM ve IREM algoritmasının tatminkar seviyedeki doğruluk oranları tarafından yansıtılmaktadır.

Sonuç olarak, yukarıda ifade edilenler çerçevesinde, mevcut bilgisayar öğrenme algoritmaları ile rekabet edebilecek seviyede doğruluk oranına sahip keREM ve IREM algoritmaları geliştirilmiştir.

1.2. Tezin İçeriği

Tezin İkinci bölümünde bilgisayar öğrenmesi üzerinde durulmuştur. Bilgisayar öğrenmesinin ne olduğu, önemi ve tipleri ele alınmıştır.

Üçüncü bölümde, denetimli bilgisayar öğrenmesi (supervised machine learning) mantığı içerisinde sınıflama (classification) işlemi incelenmiştir.

Dördüncü bölümde, temel sınıflama algoritmaları ayrıntılarıyla aktarılmıştır.

Beşinci bölümde, IREM algoritması ayrıntıları ile anlatılarak gerçek veri setleri üzerinde performans karşılaştırılmaları yapılmıştır.

Altıncı bölümde, keREM algoritması, bir önceki bölümde olduğu gibi, ayrıntıları ile anlatılarak gerçek veri setleri üzerinde performans karşılaştırılmaları yapılmıştır.

BÖLÜM 2. BİLGİSAYAR ÖĞRENMESİ (MACHINE LEARNING)

2.1. Giriş

Bu bölümde bilgisayar öğrenmesi (machine learning) konusu sunulacaktır. Bilgisayar öğrenmesinin ne olduğu, bilgisayar öğrenmesinin önemi ve tipleri ile alakalı bilgiler verilecektir.

2.2. Bilgisayar Öğrenmesi Nedir?

Öğrenmenin ne olduğu filozofik sorulardan biri olarak karşımıza çıkmaktadır. [5] Sözlük tanımı "çalışma, öğretim veya tecrübe ile bilgi edinme, anlamak, veya yetenek kazanma", "tecrübe ile davranışsal eğilimin modifiye edilmesi" gibi kavramları içermektedir [41]. Zoologlar ve psikologlar insan ve hayvanlardaki öğrenme biçimlerini araştırmaktadırlar. Hayvan öğrenmesi ile bilgisayar öğrenmesi arasında bazı paralellikler vardır. Bu paralellikler, bilgisayar öğrenmesindeki bir çok tekniğin geliştirilmesinde öncülük etmiştir[42].

Son zamanlarda, verileri (data) kullanarak model geliştirme yaklaşımları, biyolojik sistemtemlerin öğrenme kapasitelerinden, özellikle de insanlarınkinden ilham almışlardır [43]. Esasında biyolojik sistemler, yaşadıkları çevrenin, bilinmeyen ve istatiksel doğası ile veri kaynaklı olarak başetmektedir. Bebekler yürümesini öğrendiklerinde mekaniğin kanunlarının farkında değillerdir, ve birçok yetişkin arabasını kullanırken, fizik kanunlarını bilmeden sürmektedir. İnsanlar ve hayvanlar, yüzleri, sesleri ve kokuları tanıma gibi konularda ileri düzey desen tanıma kapasitesine sahiptirler. İnsanlar bu yeteneklerle doğmazlar, fakat bunları çevre ile olan veri kaynaklı etkileşimleri ile öğrenirler [44].

Veri örneklerinden öğrenme problemi, klasik felsefedeki genel çıkarımda bulunma ile ilişkilendirilir. Her tahmini öğrenme süreci iki aşamadan oluşur [66]:

1. Verilen bir örnekler setinden meydana gelen sistemin kurallarını öğrenme veya tahmin etme,

2. Öğrenilen veya tahmin edilen kurallarla, sistemin gelecekteki giriş değerleri ile çıktılarının tahmin edilmesi.

Bilgisayarda bir problemi çözebilmek için algoritmaya ihtiyaç duyulur. Algoritma girişleri (input) çıkışlara (output) dönüştürme işini yapması gereken talimatlar dizilişidir. Mesela, sıralama yapmak için algoritma tasarlanabilir. Giriş bir sayılar dizisi iken çıkış bu sayıların düzenli listesidir. Sıralama işlemi için, bir çok farklı algoritma geliştirilmiştir. Bunlar içerisinden, en az komut, hafıza veya her ikisinide ihtiyaç duyarak görevini yapan algoritmalar tercih edilir. Ancak, bazı işler için bu tarzda bir algoritma geliştirmek mümkün değildir [45]. Örneğin, DNA'daki promoterların bulunması. DNA dizisi içerisinde protein sentezleyen bölgeler (promoter) bulunmaktadır. Protein sentezleyen bölgelerin tespit edilmesi bioinformatik açısından kritik konulardan bir tanesidir. Ancak, bu bölgelerin tespit edilmesi, sayıların dizilişini gerçekleştiren tarzda bir algoritma geliştirmek kadar kolay değildir. Çünkü, bu konuda bilgisayarın öğrenmeye ihtiyacı vardır [46].

Diğer bir ifadeyle, bilgisayarın yapılan iş için otomatik olarak yol bulması istenmektedir. Sayıları sıralarken öğrenmeye ihtiyaç yoktur, bunun için zaten yapılması gereken komut sırası bellidir. Fakat, algoritması olmayan ancak sadece örnek verileri olan bir çok uygulama vardır [47][89].

Veri setleri içerisinde, ilgili disiplin ile alakalı, işe yarar, çok önemli desenler (patttern) bulunmaktadır [48][49]. Bu desenleri ortaya çıkaracak yöntem tamamen tanımlanamayabilir. Ancak iyi ve kullanışlı bir yaklaşım oluşturulabilinir. Bu kavram bilgisayar öğrenmesinin ilerlemesi gereken yönü göstermektedir. Geliştirilen yaklaşımlar herşeyi açıklayacaktır, fakat verinin önemli bir kısmının izahını yapacak, süreci anlamaya yardımcı olacak veya ileriye dönük tahmin yürütmede kullanılabilecektir [45].

Bilgisayar öğrenmesi tekniklerinin büyük veritabanlarına uygulanmasına veri madenciliği (Data Mining) denilmektedir [50][51]. Veri madenciliğinin bir tanımı da şu şekilde verilebilir; daha önceden bilinmeyen, geçerli ve uygulanabilir bilgilerin geniş veri tabanlarından elde edilmesi ve bilgilerin işletme kararları verilirken kullanılmasıdır [52].

Burada altı çizilmesi gereken noktalardan biri, elde edilecek bilginin önceden bilinmiyor olmasıdır. Veri madenciliği sonucunda ulaşılacak bilginin önceden bilinmiyor olmasında kasıt, elde edilecek sonucun önceden tahmin edilemiyor olmasıdır [52]. Ayrıca veri madenciliği, tahmin edilen, öngörülen veya başka yöntemlerle elde edilmiş sonuçların ispatının yapmak için kullanılacak bir araç değildir. Bununla beraber veri madenciliği, daha önce hiç akla gelmemiş, düşünülmemiş sonuçları ortaya koyduğundan diğer yöntemlerden farklılık gösterir [53][54].

Ancak, bilgisayar öğrenmesi sadece bir veritabanı problemi değildir, aynı zamanda yapay zekanın bir parçasıdır [55]. Zeki olmak için, değişen çevredeki sistemin öğrenebilme yeteneğine sahip olması gerekir. Eğer sistem öğrenebilir ve bu değişikliklere adapte olabilirse, tasarımcının ileriyi görmesi ve muhtemel tüm durumlar için çözüm sağlaması gerekmez. Bilgisayar öğrenmesi görüntü, ses tanıması ve robotikteki bir çok problemle alakalı çözümler sağlar [56][57]. Yüzleri tanıma örneğini ele alınacak olursa; İnsanoğlu, bu işi herhangi bir çaba harcamadan yapar, yüzlerine bakarak veya pozlardaki, ışıktaki, saç stillerindeki vb. farklılıklara rağmen resimlerinden her gün aile üyelerini ve arkadaşlar tanınabilir. Bu bilinçsizce yapılır ve nasıl yapıldığı açıklanamayabilir. Bunun sebebi kişinin kendi uzmanlığını açıklayamamasıdır. Kişi aynı şekilde bilgisayar programını yazamaz. Aynı zamanda, yüz görüntüsünün rastgele seçilmiş pixeller kolleksiyonu olmadığı bilinmektedir; yüzün bir yapısı vardır. Simetriktir. Yüzde belli yerlere yerleştirilmiş gözler, burun, ağız vardır. Herbir kişinin yüzü bu şekilde belli kombinasyonlardan oluşan bir desendir. Bir kişinin örnek yüz görüntüsünü analiz ederek, öğrenme programı kişiye özel deseni yakalar ve daha sonra da verilen görüntüdeki bu deseni kontrol ederek kişiyi tanımlar. Bu işlem desen tanımanın bir örneği olarak verilebilir [45].

Sonuç olarak, bilgisayar öğrenmesi, performans ölçütlerini optimum seviyeye çıkartmak için örnek veri veya eski tecrübeleri kullanarak bilgisayarın proglamlanmasıdır. Bu işlem için, bazı parametrelere göre tanımlanmış bir model mevcuttur. Bu çerçevede bilgisayar öğrenmesi, eğitim seti veya geçmiş tecrübeleri kullanarak model parametrelerini optimize etmek için bilgisayar programının uygulanmasıdır. Model, veriden bilgi elde etmek için geleceğe ait tahminler, açıklamalar veya her ikisini sağlayabilir [58].

2.3. Bilgisayar Öğerenmesinin Önemi

Bilgisayar öğrenmesini gerekli kılacak bir takım önemli mühendislik sebepler vardır. Bunlardan bazıları şu şekilde sıralanabilir [86].

Bazı örnekler hariç, birtakım görevler iyi tanımlanamayabilir. Yani giriş-çıkış çiftleri belirlenebilirken, girişlerle istenilen çıkışlar arasında özlü bir ilişki kurulamayabilir. Büyük çaptaki örnek girdi karşısında, doğru çıktılar üretmesi için, bilgisayarların kendi iç yapısını ayarlaması istenir. Bu yüzden örneklerde saklı ilişkileri tahmin için giriş-çıkış fonksiyonlarının oluşturulması gerekmektedir [59].

Büyük yığınlar halindeki verilerde önemli ilişkiler saklı olabilir. Bilgisayar öğrenmesi yöntemleri bu ilişkileri ortaya çıkartmak için sık sık kullanılır.

İnsan tasarımcılar, kullanıldıkları ortamda istenildiği gibi çalışmayan makinalar üretebilmektedirler. Esasında, tasarım esnasında çalışma ortamının bazı özellikleri tamamen bilinmeyebilir. Bilgisayar öğrenmesi metodları, iş esnasında varolan bilgisayarların tasarılarının geliştirilmesinde de kullanılabilir.

İnsanlar tarafından çözümlenmesi beklenen belli bir görev için varolan bilginin miktarı, aşırı büyük olabilir. Bundan dolayı, bilgisayar, bu büyük miktardaki veriden, insanın ortaya çıkaracağı bilgiden çok daha fazlasını süratle öğrenmeye muktedirdir. Zamanla çevre değişebilir. Değişen çevreye adapte olabilen bilgisayarlar, daimi olabilecek yeniden tasarımlara olan ihtiyacı azaltacaktır.

Görevlerle alakalı yeni bilgi insanlar tarafından daima keşfedilmektedir. Kelime hazinesi değişmektedir. Dünyada daima yeni olaylar akışı vardır. Sistemlerin yeni bilgilere uyumlu olması için devamlı yeniden tasarımı pratikte mümkün değildir. Fakat bilgisayar öğrenmesi metodları bu yükün çoğunun üstesinden gelebilir.

2.4. Bilgisayar Öğrenmesi Türleri

Bu kısımda, temel bilgisayar öğrenme türleri olan, denetimli, denetimsiz ve takviyeli öğrenme hakkında özet bilgiler verilecektir.

2.4.1. Denetimli öğrenme (supervised learning)

Denetimli öğrenme bilinen giriş-çıkış örneklerinden, bilinmeyen bir bağımlılığı ölçmede kullanılır. Sınıflandırma (classification) ve regresyon (regression), tümevarımsal öğrenme tarafından desteklenen yaygın işlerdir. “Denetimli” terimi, eğitim setindeki her bir örneğe ait çıktının biliniyor olduğunu göstermektedir (yani, öğretmen, danışman tarafından sağlanır) [47][50] [60].
Sınıflandırma ve regresyon, giriş X, çıkış Y’nin olduğu denetimli öğrenme problemleridir. Ve çözülmesi gereken, giriş ile çıkış arasında varolan haritanın öğrenilmesidir. Bilgisayar öğrenmesindeki temel yaklaşım şudur; bir parametreler seti ile tanımlanan bir model varsayılır,

$$y = g(x|\theta) \tag{2.1}$$

$g(\cdot)$’nin model olduğu yerde, θ onun parametreleridir. Regresyonda bir rakam olan y, sınıflandırma işleminde bir sınıf kodudur (0/1 gibi). $g(\cdot)$ regresyon fonksiyonu olduğu gibi, sınıflandırmada, farklı sınıf örneklerini ayıran ayırt edici fonksiyondur [64][45].

2.4.2. Denetimsiz öğrenme (unsupervised learning)

Denetimsiz öğrenme planı altında, sadece giriş değerleri olan örnekler, öğrenme sistemlerine verilir. Öğrenme sürecinde çıkış değerleri yoktur. Öğretmeni aradan çıkaran denetimsiz öğrenme sisteminde, öğrenicinin, modeli kendi başına oluşturması ve değerlendirmesi beklenir. Denetimsiz öğrenmenin amacı, giriş verisindeki "doğal" yapıyı keşfetmektir [64][45].

Kümeleme, en yaygın denetimsiz öğrenme işlemidir. Kümeleme, sınırlı kategori setlerinin tanınmasında veya verileri tanımlayan kümelerinin oluşturulması için gerçekleştirilen betimsel bir işlevdir [61][47].

Genomlarımızdaki DNA "yaşam planıdır" (blueprint of life) ve A, G, C, ve T isimli bazlar zincirinden oluşmaktadır. RNA, DNA dan çıkartılmaktadır ve proteinler RNA'larda çevrilmektedirler. Proteinler yaşayan vücudun kendisi ve yaptıklarıdır. DNA'nın bazlar zinciri olduğu gibi proteinde amino asidler (bazlar tarafından tanımlanan) zinciridir. Moleküler biyoloji'de bilgisayar biliminin uygulama alanlarından bir tanesi, bir diziyi diğeriyle eşleştirme olan hizalamadır (alignment). Diziler çok uzun olabileceği için, bu işlem zor bir dizi eşleştirme problemidir. Birbiriyle eşleşecek bir çok kalıp diziler vardır ve silmeler, eklemeler ve yerine koymalar olabilir. Kümeleme, proteinlerde düzenli olarak mevcut bulunan amino asitler dizilerinin oluşturduğu motifleri öğrenmek için geliştirilen metodlarda kullanılır. Eğer amino asitler harfler ve proteinler cümlelerse, motiflerde kelimelerdir; farklı cümleleri oluşturan belli anlamı olan harfler dizisidir. Bundan dolayı, DNA içerisindeki anlamlı cümleleri okuyabilmek için kelimelerin tespit edilmesi önemli bir işlemi ifade etemektedir [59][62].

2.4.3. Takviyeli öğrenme (reinforcement learning)

Bazı uygulamalarda, sistemin çıkışı bir hareketler dizisidir. Böyle bir durumda, tek bir hareket önemli değildir; önemli olan amaca ulaşmak için doğru hareketleri barındıran politikadır. Herhangi bir ara durum için en iyi hareket olarak ifade

edilebilecek bir şey yoktur; eğer iyi bir politikanın parçası ise hareket iyidir. Böyle bir durumda, bilgisayar öğrenmesi programı politikaların iyiliğini değerlendirebilir olmalıdır ve politika oluşturmak için geçmiş iyi hareket dizilerinden öğrenebilmelidir. Bu öğrenme metodlarına takviyeli öğrenme algoritmaları denilmektedir [45][47][77].

Verilebilecek iyi bir örnek te tek bir hareketin kendi başına bir öneminin olmadığı bir oyundur; iyi olan doğru hamleler dizisidir. Eğer iyi oynama politikasının bir parçası ise hamle iyidir. Hem yapay zekada hemde bilgisayar öğrenmesinde oyun oynama önemli bir araştırma alanıdır. Bunun sebebi oyunların betimlenmesinin kolay olması ve aynı zamanda iyi oynamasının epey zor olmasıdır. Satranç gibi bir oyunun az sayıda kuralı vardır ancak her durumda çok fazla oranda hamle olabileceğinden ve oyunun çok büyük hamleler içerdiğinden dolayı çok koplikedir. Bundan dolayı, oyunu iyi oynamayı öğrenebilecek iyi bir algoritmaya sahip olmak önem arzetmektedir [45][47][62].

BÖLÜM 3. DENETİMLİ BİLGİSAYAR ÖĞRENMESİ SINIFLANDIRMA

3.1. Giriş

Krediye müracaat edenlerin hangilerinin “güvenli” ve hangilerinin banka için “risk” olduğuna karar verebilmek için bankanın kredi memuru verilerini inceler. Bir pazarlama müdürü, belirli bir profildeki müşterinin bilgisayar alıp almayacağı konusunda tahmin yürütebilmek için veri analizlerine ihtiyaç duyar. Hastanın üç spesifik muayeneden hangisini alması gerektiğine karar vermek için, tıp araştırmacısı göğüs kanseri verilerini analiz etmek ister [78]. Verilen bu örneklerin her birinde gerçekleştirilen veri analiz operasyonunda kullanılan model veya sınıflandırıcı bir takım kategorik etiketlere sahiptir. Bu kategoriler şu şekilde olabilir; kredi müracaat verisinde “güvenli” veya “riskli”; pazarlama verisinde “evet” veya “hayır”; tıpbi bir veride “muayne A” “muayne B” veya “muayne C”. Bu kategoriler, değerler arasında sıranın bir anlam taşımadığı değerlerle temsil edilirler. Örneğin, 1,2 ve 3 değerleri, muayne A,B ve C’yi temsil etmekte kullanılabilir ki, bu grup arasında bir sıralama söz konusu değldir. [63].

Denetimli bilgisayar öğrenmesi (tümevarımsal sınıflandırma algoritması olarak ta bilinir) özel örneklerden genel hipotez üretmeye yarayacak algoritma arama olarak tanımlanabilir ki, bu algoritma vasıtasıyla gelecek örnekler için tahminlerde bulunulmaktadır. Diğer bir ifadeyle, denetimli öğrenmenin amacı, sınıf etiketlerinin dağıtımları için özlü bir model oluşturmaktır. Bu durmda ortaya çıkan sınıflandırıcı, örneği oluşturan karakteristiklerin bilindiği, ama sınıf etiketi değerinin belli olmadığı örneklere uygulanarak, uygun sınıf değerleri tespit edilir [58][64].

3.2. Sınıflandırma

f'nin bir fonksiyon olduğunu varsayıldığında, öğrenen bir sistemin görevi bu fonksiyonun ne olduğunu tahmin etmektir. Öğrenilecek fonksiyonla alakalı hipotez h ile belirtilir. Hem f hem de h, n bileşenli vektör-değerli girişin ($X=(x_1, x_2, ..., x_i, ..., x_n)$) fonksiyonlarıdır. Burada h, girişi X, çıktısı h(X) olan bir araç tarafından uygulandığı düşünülür. Hem f hemde h vektör-değerli olabilirler. Bir hipotez olan fonksiyon h, H fonksiyonlar sınıfından seçildiği varsayılır. f fonksiyonu, bu sınıfa ya da bu sınıfın bir alt setine aittir. Böylece, h hipotezi, m adet giriş örneklerinden oluşan eğitim setine dayanılarak seçilir. [64].

Bir A örnek setinde e adet örnek ve s adet $\{d_1,d_2,\ldots,d_s\}$ farklı sınıf olsun. $e(d_i)$, d_i sınıfına ait bir örneği göstersin. A'dan bilgi veya kavram öğrenme [65][66];

$$h:\{Des(d_i)\}\ 1\leq\ i\leq s \tag{3.1}$$

olarak gösterilir. Burada Des(di), di sınıf kavramının tanımıdır.

Herhangi bir tümevarımsal (endüktif) öğrenme prosedürü tarafından tahmin edilebilen iki şart vardır. Bunlar, eksizlik şartı ve tutarlılık şartıdır.

Eksiksizlik:

$$\forall e(d_i)\in\ A(Des(d_i)\Rightarrow e(d_i)) \tag{3.2}$$

Bu şart ile bütün örneklerde, d_i sınıfı $Des(d_i)$ kavramı tarafından kapsanmalıdır. Yani, A örnekler setindeki herbir örnek, sınıf tanımlamalarını yapan hipotezler kümesi ile mutlaka sınıflandırılmalıdır. A eğitim setinde sınıflandırılmamış hiçbir örnek kalmamalıdır.

Tutarlılık:

$$\forall e(d_i),e(d_j)\in A(e(d_j)\Rightarrow\approx Des(d_i))\ eğer\ j\neq i \tag{3.3}$$

Yani, Des(di) kavramı, di sınıfına ait olmayan herhangi bir örneği kapsamamalıdır. Diğer bir ifadeyle, A eğitim setindeki bir örnek sadece ve sadece bir sınıfa ait olacak şekilde sınıflandırılmalıdır.

Bir öğrenme işleminde, tümevarımsal öğrenme ile birkaç mümkün çıktı elde edilmiş olacaktır. **H** bütün mümkün çıktıları içeren bir hipotez uzayı olsun. Tümevarımsal öğrenme, bir **h** hipotezi bulmak için **H** hipotez uzayında bir arama prosedürü olarak açıklanabilir [67][68].

U örnekler uzayı olsun. **U**'daki örnekler bir şart karakteristik seti C={C1,C2,…,Cn} ve bir D karar karakteristiği tarafından tanımlanır. Şart karakteristiklerinin bilgi alanı V tarafından verilir (V={V1,V2,...,Vn}). Herbir Ci∈C olan Ci şart karakteristiğinin bilgi alanı Vi değer setidir. Vi, Vi∈V olan gözlemlenebilir bir değer setidir. H'nin boyutu veya mümkün hipotezlerin sayısı;

$$|H| = \sum_{i=1...n} |V_i| + \sum_{\substack{i,j=1...n \\ i \neq j}} |V_i||V_j| + ... + \sum_{\substack{i,j,...,k=1...n \\ i \neq j \neq ... \neq k}} |V_i||V_j||V_k| + \prod_{i=1...n} |V_i| \tag{3.4}$$

olarak hesaplanır. Şart karakteristikleri ikili değerlere sahip ise $|H|$;

$$|H| = \sum_{i=1...n} C_n^i 2^i \tag{3.5}$$

olur. Burada açıkça görülüyor ki, n çok büyük bir değer olduğunda bütün hipotez uzayını araştırmak pratik hatta mümkün değildir [69].

Sınıflandırma problemi metodları şu şekilde açıklanmıştır [60]:

Bir giriş örneği X= (x1, x2, ... , xd) J guruplarundan (sınıflarından) C1,C2, ...,CJ birine (ama sadece birine) sınıflandırılacaktır. Grupların varlığı öncellik (priori) olarak bilinmektedir. Giriş örneği x, genellikle sınıf üyeliği bilinmeyen nesnenin özelliklerini temsil eder. Kategorik değişken y, nesnenin sınıf üyeliğini göstersin, böylece y=j, onun Cj sınıfına ait olduğu anlamına gelir. Sınıflandırma, sınıf-üyeliği y ile, özellik vektörü x arasındaki ilişkiyle ilgilenir. Özet olarak, amaç, sınıfları belli

örneklerden oluşan eğitim seti (xi,yi), (i=l....n) kullanarak, giriş çıkış arasındaki haritanın belirlenmesidir, x → y. Bu sınıflandırma (karar kuralı olarakta bilinir), gelecekteki örnekleri sınıflandırmada kullanılır, yani y'yi sadece vektör x sınıflandırmasını kullanarak belirlemektir. Eğitim ve gelecek verinin ikiside bağımsız ve eşit bir şekilde dağıtılan aynı (bilinmeyen) istatistiksel dağılımdan ortaya çıkan örneklerdir.

Sınıflandırma, özel bir öğrenme problemini temsil eder. Sistemin çıktısının sadece 0 ve 1 olduğu y={0,1}, ikili-sınıfllama problemi ele alındığında, öğrenen bilgisayar (bilgisayar), bir dizi indikatör (sınıflandırıcı) fonksiyonu f(x,w) uygulamaya ihtiyaç duyar. Bu problem için yaygın kullanılan kayıp fonksiyon (loss function), sınıflandırma hatasını ölçer [70],

$$L\big(y, f(x,\omega)\big) = \begin{cases} 0, & Eğer\ y = f(x,\omega), \\ 1, & Eğer\ y \neq f(x,\omega). \end{cases} \tag{3.6}$$

Bu kayıp fonksiyonu kullanarak, risk fonksiyonu,

$$R(\omega) = \int L\big(y, f(x,\omega)\big) p(x,y)\,dx\,dy \tag{3.7}$$

yanlış sınıflandırma olasılığıdır. O halde öğrenme, sadece eğitim setini kullanarak, ortalama yanlış sınıflandırma hatasını minimize eden fonksiyonları f(x,w) (sınıflandırıcı) bulma prosedürü olarak tanımlanabilir [60][71].

Bilgisayar öğrenmesi metodları birçok farklı gelenekten ortaya çıktığı için, terminolojisi eşanlamlılarla doludur. Örnek olarak, giriş vektörünün bir çok farklı isimleri vardır. Bunlardan bazıları: giriş vektörü, patern vektör, özellikli vektörü, örnek. Giriş vektörünün bileşenleri, xi, farklı şekillerde, özellikler, nitelikler, giriş değişkenleri ve bileşenler olarak adlandırılırlar. Bileşenlerin değerleri üç çeşittir: Gerçel-değerli rakamlar, ayrık-değerli rakamlar veya kategöriksel değerlerdir. Kategöriksel değerleri gösteren bir örnek olarak, öğrenci hakkındaki bilgiler olan,

sınıf, bölüm, cinsiyet, danışman gibi değerlerle temsil edilebilir. Bu durumda belli bir öğrenci şu şekildeki bir vektörle temsil edilir: (ikinci sınıf öğrencisi, tarih, erkek, Eyüp Ağgez). Buna ilaveten, kategoriksel değerler sıralı (küçük, orta, büyük gibi) veya sırasız (az önceki örnekte olduğu gibi) olabilirler. Ayrıca, tüm bu değerlerin karışımı ile de giriş vektörü oluşturulabilir [64]. Tüm bu durumlarda, giriş vektörü, karakteristikleri, değerleri ile birlikte sırasız şekillerde listeleyerek temsil etmek mümkündür. Karakteristik-değer ikililerinin gösterimine bir örnek olara, (Bölüm: Tarih, cinsiyet: erkek, sınıf: ikinci sınıf, danışman: Eyüp Ağgez, yaş:19) verilebilir [72].

Çıktı kategorik bir değer olabilir. Bu durumda, içinde h hipotezinin bulunmasını içeren süreç, sınıflandırıcı (classifier), tanıyıcı (recognizer), veya kategori edici (categorizer) olarak isimlendirilir. Çıktı ise, sınıf, kategori veya karar olarak adlandırılır. Sınıflandırıcıların bir çok problemde uygulamaları vardır, mesela; el yazımı karakterleri tanımlama işlemi. Bu durumda giriş yazılı karakterlerin uygun temsilleridir, ve sınıflandırıcı, girişleri 64 kategoriden birisine koyar. [64][73].

3.3. Denetimli Bilgisayar Öğrenmesi Sınıflandırma Yöntemleri

Denetimli bilgisayar öğrenmesi sınıflama konusunda bir çok algoritma geliştirmiştir. Bir sonraki bölümde ayrıntılı olarak ele alınacak olan algoritmalar aşağıdaki şekilde kategorize edilebilir.

3.3.1. Mantıksal/sembolik sınıflandırıcılar

Mantıksal/Sembolik sınıflandırıcı kategorisinde iki yaklaşım bulunmaktarı; karar ağaçları ve kural tabanlı sınıflandırıcılar [58].

3.3.1.1. Karar ağaçları

Karar ağaçlarında hedef fonksiyonu bir karar ağacı olarak tanımlanmaktadır. Karar ağacı öğrenimindeki aramalar, karakteristik hakkındaki bir denemenin ne kadar bilgi vereceğini bildiren entropi tabanlı bilgi elde etme ölçümü tarafından yönlendirilir.

Öğrenme algoritmalarının genellikle küçük ağaçlara karşı yatkınlıkları vardır. Bu yöntem, istekli, denetimli ve değişken bir metoddur. Gürültülü veriye duyarlı, çoklu parametrelere uygundur. Öğrenme sürecinde ön bilgi gerektirmez. Ancak, geniş verilerle bir çok farklı yollarla iyi sonuç verir [94][61].

3.3.1.2. Kural tabanlı sınıflandırıcılar

EĞER-İSE kuralları, ilk olarak bir karar ağacı oluşturmayı gerektirmeden, sıralı kapsayan algoritma(sequential covering algorithm) kullanarak eğitim verilerinden elde edilebilir. Bu yaklaşım ismi, sırayla öğrenilen kurallar kavramından gelmektedir. Verilen bir sınıf için herbir kural, ideal olarak aynı sınıfa ait bir çok örneği kapsayıp diğer sınıflara ait hiçbir örneği kapsamıyacaktır [85].

3.3.2. Bayes sınıflandırıcıları

Bayes öğrenimi, ilişki seviyesinin olasılık dağılımlarıyla belirlendiği bir varsayıma dayanan olasılıkçı bir yaklaşım sunmaktadır. Optimal karar veya sınıflandırmalara, incelenen verilerin yanısıra bu olasılıkların değerlendirmesiyle ulaşılabilir. Bayesci öğrenim metodları öğrenim sisteminden elde edilen çıktılar esas alınarak iki gruba ayırılabilir: belirli eğitim verileriyle mümkün olan en iyi hipotezi elde edenler ve yine belirli bir eğitim setiyle verilen yeni örneğin mümkün olan en iyi sınıflandırmayı üretenler. Bu nedenle hedef fonksiyonu birinci grupta açık olarak temsil edilir fakat ikinci grupta dolaylı olarak tanımlanır. Temel avantajlarından birisi de, önceki bilgi ile uyum sağlamasıdır. Görünmeyen bir örneğin sınıflandırması, çoklu hipotezlerin tahminlerinin birleştirmesi yoluyla elde edilir [74][75][76].

Kapsamlı verilerle de iyi sonuç vermektedir. İstekli, denetimli bir öğrenme metodudur ve öğrenim sürecinde araştırma gerektirmez. Bayesci öğrenimin gürültülü verilerle problemi olmasada küçük veri setleriyla zorlukları vardır. Bayesci öğrenim, en az tanımlama uzunluğu ilkesine dayanan bir yaklaşımı benimser. [77][78].

3.3.3. Tembel-öğrenme (lazy learning) algoritmaları

Örneğe dayalı (instance-based) öğrenim olarakta isimlendirilir. Eğitim verilerinin ötesinde görünmeyen olayın sınıflandırılmasına kadar genelleştirilmesi anlamında tipik bir tembel öğrenme yaklaşımıdır [79]. Ayrıca, hedef fonksiyonu açık olarak tanımlanmamıştır; bunun yerine, öğrenen sistem, belirli görünmeyen olayı sınıflandırırken hedef fonksiyon değerini geri verir. Hedef fonksiyon değeri bütün eğitim setine dayanmaktan ziyade görünmeyen olaya ait olarak kabul edilen eğitim verilerinin bir alt kümeye dayanmasıyla elde edilir. Bu, farklı hedef fonksiyonunun, belirgin görünmeyen örneğe benzemesi anlamına gelir. Tek bir hedef fonksiyonunun, öğrenen sisteminin tüm eğitim verilerini genelleme yapması sonucunda elde edilmesi yaklaşımını kullanan öğrenme yöntemlerinden belirgin bir şekilde farklıdır. Bu arayış süreci istatistiki anlayışa dayalıdır ve belirli görünmeyen olaya yakın eğitim verilerinin belirlenmesi ve komşularına dayanarak hedef fonksiyon değeri elde edilmesinden oluşmaktadır. K-en yakın komşular (K-nearest neighbors), olaya-dayalı tümevarım (case-based reasoning), ve yerel ölçülmüş regrasyon, bu kategorideki en popüler algoritmalardır [80][81][82].

3.3.4. Yapay sinir ağları

Sinir ağı öğreniminde, verilen bir sabit ağ yapısında, hedef fonksiyonu öğrenme ağ ağırlıklarının bulunmasıyla eş anlamlıdır. Öyleki, ağ çıktıları, eğitim verilerinde belirtildiği gibi beklenen sonuçlarla aynı veya kabul edilebilir bir aralıktadır [83][84]. Esas itibariyle bir ağırlık vektörü hedef fonksiyonu tanımlar . Bu, hedef fonksiyonun insanlar tarafından okunmasını ve yorumlanmasını çok zorlaştırır. Yapay sinir ağları , istekli, denetimli ve dengesiz bir öğrenim yaklaşımıdır. İleri-besleme ağları için bir popüler algoritma, geri yayılımdır [85].

3.3.5. Destek vektör makinası (support vector machines)

Destek vetkör makinaları, doğrudan giriş alanındaki verilerden lineer olmayan hedef fonksiyonu öğrenmesi yerine, ilk olarak, eğitim verilerini giriş alanından yüksek

boyutlu özellik F alanına dönüştürmek için eğitim verilerinin iç ürünü şeklinde tanımlanan bir çekirdek fonksiyonu kullanır. Sonra, F'deki optimal lineer ayırıcıyı (bir hiper düzlem) öğrenir. Lineer ayırıcıya dayanarak tanımlanan karar fonksiyonu, daha önce görünmeyen olayları sınıflandırmak için kullanılabilir. Çekirdek fonksiyonu, destek vektör makinalarında merkezi bir rol oynar. Çekirdek fonksiyonu destek vektörleri sadece verilen eğitim verilerinin bir alt kümesine dayanmaktadır [86][87][88].

3.4. Doğruluk ve Hata Ölçüleri

Bir sınıflandırıcı eğitildiğinde birçok soru ön plana çıkabilir. Örneğin, sınıflandırıcıyı müşteri satınalma davranışını tahmin etmek için eğitirken, önceki satışlardan elde edilen verilerin kullanıldığı farzedilsin. Eğitim setinde yer almayan müşteri verileriyle eğitilen sınıflandırıcının, gelecekteki müşterilerin satınalma davranışlarını ne kadar doğru tahmin ettiği hakkında bir değerlendirme istenebilir. Birden fazla sınıflandırıcı geliştirmek için farklı yöntemler denenip onların doğruluğunun karşılaştırmak istenebilir.

Doğruluğun ne olduğu nasıl tahmin edilebilineceği ve öğrenilen bir modelin doğruluğunu artırmak için stratejiler tartışılacaktır.

3.5. Sınıflandırıcı Doğruluk Ölçüleri

Eğitim verilerini kullanarak sınıflandırıcı elde etmek ve ortaya çıkan modelin doğruluğunu tahmin etmek algoritmanın verilere karşı aşırı uzmanlaşmasından dolayı yanıltıcı, fazla iyimser tahminler yapılmasıyla sonuçlanabilir. Bunun yerine doğruluk, daha önceden model eğitiminde kullanılmayan sınıf-etiketli örnekler takımından oluşan testle daha iyi ölçülür. Belirli test takımındaki sınıflandırıcının doğruluğu, sınıflandırıcı tarafından doğru olarak sınıflandırılan test takım örneklerinin yüzdesidir [76][85]. Resim tanıma literatüründe, sınıflandırıcının genel tanıma oranı olarak da bahsedilmektedir, yani bu, sınıflandırıcının değişik sınıflardaki örnekleri ne kadar iyi tanıdığını yansıtır [89].

M’nin bir sınıflandırıcı olduğu farzedilirse, Acc(M), M’nin doğruluğuyken, hata oranı veya yanlış-sınıflandırma oranı 1-Acc(M)’dir. Eğer eğitim seti, bir modelin hata oranını tahmin etmek için kullanılırsa, hesaplanan değer tekrar-yerine koyma oranı olarak bilinir. Bu hata tahmini, doğru hata oranının çünkü model daha önce görüldüğü örneklerin hiçbirinde denenmemiştir [90].

Karışıklık matrisi (confusion matrix), sınıflandırıcının farklı sınıflardaki örnekleri ne kadar iyi tanıdığını analiz etmede kullanılan faydalı bir araç olarak kabul edilir. İki sınıf için bir karışıklık matris Şekil 3.1’de gösterilmiştir. Belirli m sınıflarında, karışıklık matrisi en az mXm boyutunda bir tablodur. CMi girdisinde, ilk m dizisindeki ve m sütunlarındaki j sınıflandırıcı tarafından j sınıfı olarak etiketlendirilen i sınıfının örnek sayısını belirtir. Sınıflandırıcının yeterli doğruluğa sahip olması için, CM1 girdisinden, 1’den CMm, m’ye kadar, geri kalan girdiler sıfıra yakınken, ideal olarak örneklerin çoğunluğu karışıklık matrisinin çaprazından gösterilecektir. Tablonun her sınıfın toplamlarını ve tanınma oranlarını sağlayacak ilave dizileri ve sütünları olabilir [85][91].

İki sınıf göz önünde tutulursa, pozitif örnekler (temel ilgi sınıfı örnekleri, mesela, bilgisayar satın alır = evet)’e karşın negatif örnekler (mesela, bilgisayar satın alır=hayır) açısından belirtilebilir. Doğru pozitifler sınıflandırıcı tarafından doğru olarak etiketlenen pozitif örneklerle ifade edilirken, doğru negatifler sınıflandırıcı tarafından doğru olarak etiketlenen negatif örneklerdir. Yanlış pozitiflerde yanlış olarak etiketlenen negatif örneklerdir. (mesela, sınıf örneklerinde bilgisayar satın alır=hayır sınıflandırıcı tarafından bilgisayar satın alır=evet olarak tahmin edilmiştir) [85][90][91].

		Tahmin edilen sınıf	
		C1	C2
Gerçek Sınıf	C1	doğru pozitifler	yanlış negatifler
	C2	yanlış pozitifler	doğru negatifler

Şekil 3.1 Bir karışıklık matrisi için pozitif ve negatif demetleri

Bunun gibi, Yanlış negatiflerde yanlış olarak etiketlenen pozitif örneklerdir. (mesela, sınıf örneklerinde bilgisayar satınalır=evet sınıflandırıcı tarafından bilgisayar

satınalır=hayır olarak tahmin edilmiştir). Bu terimler sınıflandırıcının kabiliyetini analiz ederken faydalıdır.

Doğruluk ölçümlerine alternatifler de vardır. Bir sınıflandırıcının tıbbi veri örneklerini "kanser" veya "kanser değil" olarak sınıflandırmak için eğitildiği farzedilsin. Bir doğruluk oranı, 90% sınıflandırıcıyı oldukça doğru olarak gösterirken, eğitim örneklerinin 3–4%'ü gerçekte "kanser" olduğunda, 90% doğruluk oranı kabul edilir olmayacaktır. Duyarlılık ve özgünlük ölçüleri sırasıyla bu amaç için kullanılabilir. Duyarlılık doğru pozitif (tanıma) oranı olarak da ifade edilirken (yani, doğru olarak teşhis edilen pozitif örneklerin miktarı), özgünlük, doğru negatif oranı (yani, doğru olarak teşhis edilen negatif örneklerin miktarı) dır [76][91].

İlave olarak, hassasiyet kullanilarak "kanser" olarak etiketlenen gerçek "kanser" olan bu ölçüler şu şekilde tanımlanmıştır [85][90][91]:

$$belirlilik = \frac{t\_neg}{neg} \tag{3.8}$$

$$hassaslık = \frac{t\_pos}{pos} \tag{3.9}$$

$$kesinlik = \frac{t\_pos}{(t\_pos + f\_pos)} \tag{3.10}$$

tpoz doğru pozitiflerin sayısı iken (aslında doğru olarak sınıflandırılan "kanser" örnekleri), poz pozitif ("kanser") örneklerin sayısı, tneg gerçek negatiflerin sayısı (aslında doğru olarak sınıflandırılan "kanser değil" örnekleri), neg negatif ("kanser değil") örneklerin sayısı, ve fpoz yanlış pozitiflerin sayısı ("kanser" olarak yanlış

etiketlenen “kanser değil” örnekleri)’dır. Doğruluk duyarlılık ve özgünlüğün fonksiyonu olduğu gösterilebilir,

$$doğruluk = hassaslık \frac{pos}{(pos + neg)} + belirlilik \frac{neg}{(pos + neg)} \quad (3.11)$$

3.6. Sınıflandırıcının Doğruluğunun Değerlendirmesi

Bilgisayar öğrenim süreci vasıtasıyla elde edilen bir modelin amacı yeni örnekleri doğru olarak sınıflandırma/tahmin etmedir. Bir Modelin kalitesinin, genellikle kullanılan ölçüsü, kestirimci doğruluktur (predictive accuracy). Yeni örneklerin kendi öğrenim safhasında model tarafından kullanmadığı varsayıldığından, modelin kestirimci doğruluğunu, gerçek hata oranı kullanarak tahmin edilmesi gerekmektedir. Gerçek hata oranı, istatistik olarak modelin hata oranının asimtotatik olarak geniş sayıdaki yeni olaylar üzerinde bulunmasıdır. Pratikte, bilgisayar öğrenmesi modelinin gerçek hata oranı mevcut olan genel olarak eğitim ve test setleri olarak ayrılan bütün örneklerle tahmin edilmelidir. Model ilk olarak eğitim örnekleri kullanılarak tasarlanmıştır, ve sonra test örneklerindeki performansına dayanarak değerlendirilmiştir. Bu hata ölçümü gelecek model performansını tahminde güvenilir olabilmesi için eğitim ve test setlerinin sadece yeteri kadar geniş olmakla kalmamalı, bağımsızda olmalılar [93][85][92].

Mevcut örnekler, eğitim ve test setleri oluşturmak için nasıl ayrılmalıdır?[83] Eğer eğitim takımı küçükse, sonuçta oluşan model çok kuvvetli olmayacaktır ve düşük genelleme kabiliyetine sahip olacaktır. Diğer taraftan, eğer test küçükse, tahmini hata oranına olan güven düşük olacaktır. Hata oranlarını tahmin etmek için değişik yöntemler kullanılmaktadır. Bu yöntemler, mevcut örneklerin eğitim ve test seteleri olarak kullanılması hususunda farklılık göstermektedir. Eğer mevcut örneklerin sayısı son derece genişse(örneğin 1 milyon), bütün bu yöntemler muhtemelen aynı hata oranı tahminine yönlendirir. Eğer mevcut örneklerin sayısı küçükse, bilgisayar öğrenim deneylerinin tasarımcısı verileri ayırırken çok dikkatli olmalıdır. Örneklerin nasıl alt kümelere ayırılacağı konusanda iyi bir mevcut klavuz yoktur. Veri nasıl

ayrılırsa ayrılsın, farklı tesadüfi ayrılmalar hatta eğitim ve test setleri belirli boyutta olsa bile, farklı hata tahminlerinin netice verecekleri açıktır [93][94].

Veri setini, eğitim ve test olarak ayırmak için, tekrar-örnekleme metodları (resampling method) olarak isimlendirilen farklı yöntemler kulanılır. Modellerin tahmini ve seçiminde tekrar-örnekleme yaklaşımının, analitik yaklaşım kullanımına olan avantajı, eski verilerin istatistiki dağılımı veya yakınsayan fonksiyonların belirli özelliklerinde varsayımlara dayanmamasıdır. Tekrar-örnekleme tekniklerinin temel dezavantajları onların yüksek hesaplama sayıları ve tekrar-örnekleme stratejisine dayanan tahminlerdeki değişmelerdir [83][85][93].

Model tahmininde temel yaklaşım, ilk olarak veri setinin bir parçasını kullanarak bir model hazırlamak veya keşfetmek ve sonra geri kalan örnekleri kullanarak bu model için risk öngörü tahmini yapmaktır. Verinin ilk parçasına eğitim seti denir, ve ikinci parçaya test (doğrulama) seti adı verilir. Bu basit strateji eğitim seti ve test setlerinin aynı bilinmeyen veri dağılımının temsilcileri olarak seçildikleri varsayımı üzerine dayanır. Bu, genellikle geniş veri setleri için doğrudur, fakat stratejinin küçük veri setleri için açık bir dezavantajı vardır. Küçük sayıda örneklerle, veri ayırma özel yöntemi modelin doğruluğu üzerinde etki sahibi olmaya başlar. Tekrar-örneklemenin değişik yöntemleri küçük veri setleri için kullanılır, ve ilk veri setini ayırmada kullanılan stratejiler bakımından farklılık gösterirler [92][93][94].

İlerleyen kısımlarda, günümüzün bilgisayar öğrenim uygulamasında yaygın olan tekrar-örnekleme yöntemlerinin özet bir açıklaması verilecektir. Bilgisayar sistemi tasarımcısı, veri ve problemin özelliklerine dayalı olarak bir seçim yapmak zorundadır.

3.6.1. Tekrar yerine koyma (resubstitution) yöntemi

En kolay yöntemdir. Mevcut olan verilerin hepsi hem eğitim hemde test için kullanılır. Başka bir deyişle, eğitim ve test setleri aynıdır. Bu, “veri dağılımı” için hata oranının tahminini iyimser olarak yapar (tahmini hata genellikle modelin gerçek uygulamalarında beklenilenden daha küçüktür). Bu nedenle yöntem gerçek-yaşam bilgisayar öğrenim uygulamalarında çok nadir kullanılmaktadır [83][95].

3.6.2. Anlaşmazlık (holdout) yöntemi

Verilerin yarısı, veya bazen verilerin 2/3'ü eğitim için kullanılır ve kalan veriler test için kullanılır. Eğitim ve test setleri bağımsız ve hata tahmini karamsardır. Farklı bölümlemeler farklı tahminler verecektir. Sürecin rastgele seçilen farklı eğitim ve test setleriyle tekrarı, ve hata sonuçlarının bir standart parametreye birleştirilmesi modelin tahminini geliştirecektir [83][85][92].

3.6.3. Özyükleme (bootstrap) yöntemi

Özyükleme yöntemi belirli eğitim örneklerini eşit oranda yenileyerek örneklendirir. Yani, her defasında bir kayıt seçilir, eşit olarak tekrar seçilebilmesi muhtemeldir ve eğitim setine tekrar eklenebilir. Örnek olarak, eğitim seti için rastgele örnekler seçen bir makine verilebilir [93].

Bir çok özyükleme yöntemi vardır. Yaygın olarak kullanılan, aşağıdaki gibi çalışan .632 özyüklemedir. **d** sayıda örnek barındıran bir veri seti verildiği farzedilsin. Veri seti, **d** kere yenilenerek örneklendirilmesiyle, özyükleme örneği veya **d** elemanlı eğitim setinin oluşmasıyla sonuçlanır. Muhtemelen orjinal veri örneklerinin bazıları bu örnekte birden fazla meydana gelir. Veri örnekleri, eğitim setini dönüştürmemesi test setinin oluşmasıyla biter. Bunun defalarca denenmesi gerektiği farzedilsin. Bu durumda, ortalama olarak, orjinal veri örneklerinin %63.2'si özyükleme ile sonuçlanabilir, ve geride kalan 36.8 test setini oluşturur (bundan dolayı isim, .632 özyüklemedir) [93][94].

63.2% rakamı nereden gelmektedir?[83] Her örneğin 1/d oranında seçilme ihtimali vardır, bundan dolayı seçilmeme ihtimali (1-1/d) dir. D kere seçilmelidir, bundan dolayı örneğin bütün bu zaman zarfı içinde seçilmeme ihtimali (1-1/d)d dir. Eğer d genişse, ihtimal e^{-1}=0.368.14 yaklaşır. Böylece örneklerin 36.8%'i eğitim için seçilmeyecektir ve dolayısıyla test setinde olacaktır, ve geri kalan 63.2% eğitim setini oluşturacaktır [85][93].

Örnekleme k kere tekrarlanabilinir, her tekrarlama için, mevcut test verisiyle yine şimdiki özyüklemeden elde edilen modelin doğruluk tahminini kullabilir. Genel bir modelin doğruluğu aşağıdaki gibi tahmin edilmiştir:

$$Acc(M) = \sum_{i=1}^{k} (0.632 \times Acc(M_i)\ test\_set + 0.368 \times Acc(M_i)\ eğitim\_set) \quad (3.12)$$

Acc(Mi)test-seti modelin özyüklemeyle elde edilen örneği i olduğunda bu test takımı i'ye uygulanır. Acc(Mi)test-seti orjinal veri setlerine uygulandığında modelin çizgi-atkısı i ile elde edilen doğruluğudur. Özyükleme yöntemi küçük verilerle çok iyi çalışır [93][94][95].

3.6.4. K-kere çapraz doğrulama (K-fold cross validation)

Veri seti X, rastgele K eşit boyutlu parçalara ayrılmıştır, Xi, i=1,...,K. Her çifti elde etmek için, K parçalarından birisini dışarıda test seti olarak tutulur ve geri kalan K parçalarını K – 1 eğitim setlerini oluşturmaları için birleştirilir. Bunu K kere yaparken, her defasında K parçlarından birini dışarıda bırakarak, K çiftlerini elde edilir [83][85][92][94].

$V_1 = X_1$ $T_1 = X_2 \cup X_3 \cup \ldots \cup X_K$

$V_2 = X_2$ $T_2 = X_1 \cup X_3 \cup \ldots \cup X_K$

...

$V_K = X_K$ $T_K = X_1 \cup X_2 \cup \ldots \cup X_{K-1}$

Bu yöntemde iki problem vardır. Birincisi, eğitim setini geniş tutmak için, küçük olan test setlerine izin verilir. İkincisi, eğitim setleri epeyce üst üste gelir, yani, herhangi iki eğitim takımı K-2 parçalarını paylaşır. K tipik olarak 10 veya 30 dur. K artınca, eğitim örneklerinin yüzdeside artar, ve daha kuvvetli tahminciler elde edilir, fakat test setleri küçülür. Bundan başka, sınıflandırıcıyı K kere eğitmenin masrafı da vardır, bu da K artınca artar. N artınca K küçülebilir; eğer N küçükse, K geniş olmalı ki yeterli eğitim birini-dışarıda-tut setlerine izin verilebilinir. K-kere çapraz-

doğrulamanın aşırı olaylarından birisi N veri takım örnekleri verildiğindedir, sadece bir örnek test seti olarak dışarıda bırakılmıştır ve eğitim N-1 örneklerini kullanır. Sonra her tekrarda farklı örneği dışarıda bırakarak N ayrı çiflerini elde edilir. Bu etiketli veriyi bulmanın zor olduğu tibbi teşhisler gibi uygulamalarda kulanılır. Birini-dışarıda-tut katmanlaşmaya izin vermez. Son günlerde, hesaplama işlemleri klaylaşınca, K-kere çapraz-doğrulamayı çok defa çalıştırmak mümkün oldumuştur, örneğin, 10×10-kere, ve ortalamalar kullanarak daha güvenli hata tahminleri elde edilir [92][94].

3.7. Topluluk Yöntemleri-Doğruluk Oranını Artırma

Tüm verilerde en iyi performansı gösterecek bir sınıflandırıcı yoktur (no free lunch theorem) [96]. Bir veri setinde çok iyi performans sergileyen bir sınıflandırıcı başka bir veri setinde düşük performans göstermektedir. Bundan dolayı araştırmacılar, bir veri setini sınıflandırırken birden fazla sınıflandırıcıyı kullanırlar. Birden fazla sınıflandırınının kullanımında iki metod kullanılır: Torbalama (bagging) ve yükseltme (boosting). Bu yöntemler, bilgisayar öğrenmesi algoritmalarının bir kombinasyonunu kullanırlar. Herbirisi k adet sınıflandırıcı modeller serisini, (M1, M2, … , Mk), gelişmiş birleşik bir M_modeli oluşturmak amacı ile birleştirir [83][85].

3.7.1. Torbalama (bagging)

Torbalamanın, doğruluğu artırma yöntemi olarak nasıl çalıştığına dair sezgisel bir bakış ortaya konulacaktır. Açıklamanın kolaylığı açısından, ilk olarak modelin bir sınıflandırıcı olduğu farzedilir. Hasta olan biri, belirtilerine göre tıbbi teşhis yaptırmak ister. Bir doktora sormak yerine, bir çok doktora sormayı tercih edebilir. Eğer belirli bir teşhis diğerlerinden daha fazla ortaya çıkarsa, bunu son veya en iyi teşhis olarak seçer. Yani, son teşhis her doktorun eşit oyunun olduğu çoğunluğun oylarına dayanmaktadır. Her doktor bir sınıflandırıcıyla değiştirilirse, temel torbalama fikrine sahip olunur. Sezgisel olarak, geniş doktor grubu tarafından oluşan

çoğunluğun oyları küçük gurup tarafından oluşan çoğunluk oyundan daha güvenlidir [93][97][98].

Verilen, d örnekli bir D seti için, torbalama şu şekilde çalışır. Herbir tekrarda, i (i = 1, 2, …, k) için, d örnek sayısına sahip, bir Di eğitim seti, D orjinal set verilerini değiştirilerek örneklendirilir. Torbalama terimi, aynı zamanda özyükleme kümelenmesi anlamına gelmektedir. Her eğitim takımı kısım 3.6.3'te belirtildiği gibi, bir özyükleme örneğidir. Çünkü değişimli örnekleme kullanılmıştır, D nin orjinal örneklerinin bazıları Di ye dahil edilmeyebilirken, diğerleri birden fazla meydana gelebilir. Bir sınıflandırıcı modeli, Mi, her Di eğitim takımı için eğitilmiştir. Bilinmeyen X örneğini sınıflandırmak için, her Mi sınıflandırıcısı bir oy olarak sayılan kendi sınıf tahminine geri döner. Torbalanmış M_sınıflandırıcısı oyları sayar ve sınıfı en çok oyla X'e tahsis eder. Torbalama, belirli test örneğine ait her tahmin averaj değeri alınarak devam eden değerlerin tahmininde uygulanabilir [92][75][76].

Torbalanmış sınıflandırıcı genelde orjinal eğitim verisi D'den türetilen tek bir sınıflandırıcıdan önemli derecede daha büyük doğruluğa sahiptir. Gürültülü verilerin etkilerine karşı oldukça kuvvetli olacaktır. Artırılmış doğruluk meydana gelir çünkü birleşik model bireysel sınıflandırıcıların değişkenliğini düşürür. Tahmin için, torbalanmış tahminci daima D den türetilen tek bir tahminciye karşı gelişmiş bir doğruluğa sahip olacağı teorik olarak ispatlanmıştır [74].

3.7.2. Yükseltme (boosting)

Bir önceki kısımdaki gibi, bir kişinin, hastalığa ait bazı belirtileri olduğunda, bir doktora danışma yerine bir çok doktora danışmayı tercih edebilir. Doktorların yaptığı önceki teşhislerin doğruluğuna dayanarak, her doktorun teşhisine değer veya kıymet ağırlığı tayin edildiğinde, son teşhis, ağırlıklı teşhislerin bir birleştirmesi olacaktır. Bu yükseltme yaklaşımının temelini oluşturmaktadır [95][99].

Yükseltme metodunda, her eğitim örneğine ağırlıklar tayin edilir. K sınıflandırıcılar serisi tekrarla eğitilir. Mi sınıflandırıcısı eğitildikten sonra, ağırlıklar sonraki Mi+1 sınıflandırıcısına Mi tarafından yanlış sınıflandırılan eğitim örneklerine "daha fazla dikkat etme" ye imkan vermek için güncelleştirilir. Son yükseltilmiş M*

sınıflandırıcısı her bireysel sınıflandırıcının oylarını birleştirir ki her sınıflandırıcının oyunun ağırlığı kendi doğruluğunun bir fonksiyonudur. Yükseltme algoritması devam eden değerlerin tahmini için uzatılabilir. Adaboost popüler bir yükseltme algoritmasıdır. Bazı öğrenim yöntemlerinin doğruluğunu artırılmasının istendiği farzedilsin. D, sınıf etiketli veri seti örnekleri verildiğinde, (X1, y1), (X2, y2), … , (Xd, yd), yi, Xi kayıtının sınıf etiketidir. Başlangıçta, Adaboost her eğitim örneğine eşit 1/d ağırlığı tahsis eder. Topluluk için k sınıflandırıcıları oluşturmak için kalan algoritma boyunca k turlarına ihtiyaç vardır. i turunda, D'den örnekler d boyutunda Di eğitim seti oluşturmak için örneklendirilmiştir. Değiştirerek örneklendirme –aynı örnek birden fazla seçildiğinde kullanılır. Her örneğin seçilme şansı ağırlığına bağlıdır. Mi sınıflandırıcı modeli Di'nin eğitim örneklerinden türetilmiştir. Hatası sonradan Di test takımı olarak kullanılarak hesaplanır [85][92][75].

Eğitim örneklerinin ağırlığı sonradan nasıl sınıflandırıldıklarına göre ayarlanır. Eğer bir örnek yanlış sınıflandırılırsa ağırlığı artar. Eğer doğru sınıflandırılırsa ağırlığı azalır. Örneğin ağırlığı, onu sınıflandırmanın ne kadar zor olduğunu yansıtır – ağırlık yüksek olursa, yanlış sınıflandırılması daha sık olur. Bu ağırlıklar bir sonraki turda, sınıflandırıcısı için eğitim örnekleri oluşturmak için kullanılır. Temel fikir, bir sınıflandırıcı tasarlandığında, onun önceki turdaki yanlış sınıflandırılmış örneklere odaklanması istenir. Bazı sınıflandırıcılar kimi "zor" örnekleri sınıflandırmada diğerlerinden daha iyi olabilirler. Bu şekilde, birbirini tamamlayan sınıflandırıcılar serisi yapılır [93][85][75].

Algoritmada kullanılan matematik temeller şu şekildedir. Model Mi'nin hata oranını hesaplamak için, Mi'nin yanlış sınıflandırdığı Di'deki her bir örneğin ağırlığı toplanır,

$$hata(M_i) = \sum_j^d w_j \times err(X_j) \tag{3.13}$$

err(Xj), Xj örneğinin yanlış sınıflandırma hatasıdır. Eğer örnek yalnış sınıflandırılmışsa, o zaman err(Xj) 1 dir. Aksi takdirde, 0 dır. Eğer Mi sınıflandırıcısının performansı zayıfsa hatası 0.5 i geçer, ve terkedilir. Yerine, yeni bir Di eğitim takımı oluşturması denenir ve ondan da yeni bir Mi türetilir. Mi nin hata oranı eğitim örneklerinin ağırlıklarının nasıl güncelleştiğine etki eder. Eğer i

turundaki bir örnek doğru olarak sınıflandırılmışsa, ağırlığı err(Mi)/(1-err(Mi)) ile çarpılır. Doğru sınıflanrılmış bütün örneklerin ağırlıkları güncelleştirildiğinde, tüm örneklerin ağırlıkları (yanlış sınıflandırılmışlar da dahil) normal hale gelir böylece toplamları daha önce olduğu gibi kalır. Bir Ağırlığı normalleştirmek için, yeni ağırlıkların bölümünün toplamı eski ağırlıklarının toplamıyla çarpılır. Sonuç olarak, yanlış sınıflandırılmış örneklerin ağırlıkları artar ve doğru sınıflandırılan örneklerin ağırlıkları yukarıda belirtildiği gibi azalır [75][77].

Yükseltme işlemi bittiğinde X örneğinin sınıf etiketini tahmin etmede sınıflandırıcılar topluluğu şu şekilde kullanılır; her sınıflandırıcıya eşit oy tahsis edildiği torbalamanın aksine, yükseltme yaklaşımında, sınıflandırıcının nasıl yaptığına dayanarak her sınıflandırıcının oyuna bir ağırlık tahsis edilir. Sınıflandırıcının hata oranı düşük olursa, daha eksiksiz olur, ve bundan dolayı, oy için ağırlığı daha yüksek olmalıdır. Mi sınıflandırıcısının oyunun ağırlığı,

$$log\frac{1-error(M_i)}{error(M_i)} \tag{3.14}$$

ile verilir. Her c sınıfı için, c sınıfından X'e tahsis edilen her sınıflandırıcının ağırlığı toplanır. En yüksek toplamı olan sınıf "kazanan" dır ve örnek X'in sınıf tahmini olarak geri döner [83][85][75][77].

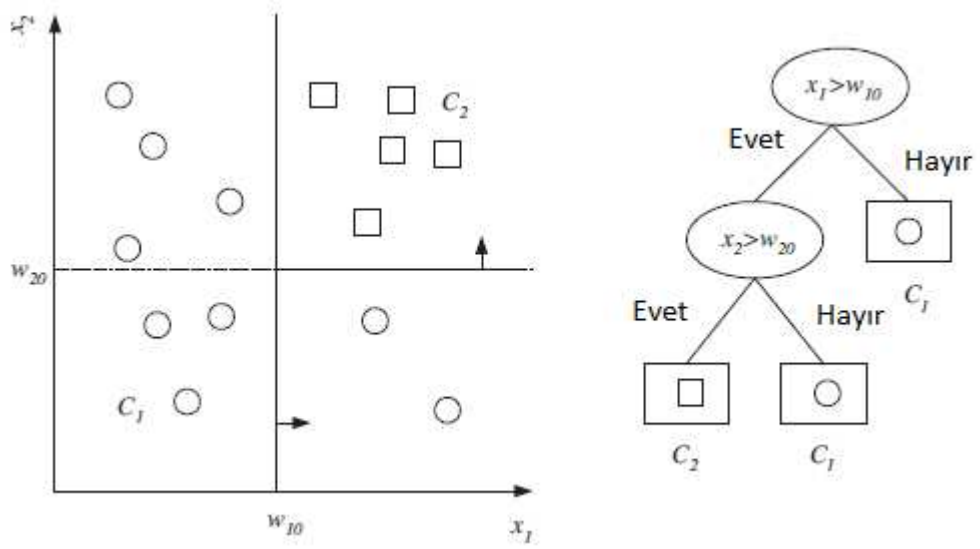
BÖLÜM 4. DENETİMLİ SINIFLANDIRMA ALGORİTMALARI

4.1. Karar Ağaçları

Karar ağacı algoritması, bilgisayar öğrenmesinde sınıflandırma algoritmalarından bir tanesidir ve bilgi teorileri ilkelerine dayanmaktadır. Bir karar ağacı algoritması otomatik olarak yayılabilen modeller oluşturur. Kullanıcı tarafından kolayca yorumlanabilir, bilinmeyen ve gürültülü verilerle uğraşır [100].

Bir karar ağacı, karar ağaçlarının yapısını temsil eden iç düğümler, dallar ve yaprak düğümlerinden meydan gelir. Ağacın üst düğümüne kök düğümü denir; iç düğümler karakteristik değerlerinde yapılan testleri temsil eder; dallar denemelerdeki farklı sonuçları; ve yaprak düğümleri düğüme düşen örneklerin sınıflandırılmasını temsil eder [92][74][77].

Karar ağaçlarının sonuçta oluşturdukları modeller, insan analizciler tarafından rahatça anladıkları için, veri madenciliği çevrelerinde özellikle ilgi çekicidir. Karar ağaçlarının inşaası analizcinin girdi parametreleri sağlamasını gerektirmez; veri hakkında önceden bilgi sahibi olunmasına da gerek yoktur. Bir kayıtın eşsiz bir yaprak düğümüyle ilişkilendirilmesi kökle başlayarak ve ayırıcı kritere dayanarak tekrar tekrar aktif düğümde girdi kayıtları üstündeki şartları değerlendiren bir çocuk düğümünü seçmeyle mümkün olabilir [85]42[92].



Şekil 4.1. Karar ağacı oluşumu

Karar ağacı oluşturma algoritmaları iki etaptan meydan gelir; ağaç inşası ve budama. Bir öncekinde, karar ağacı inşa algoritmalarının çoğunda ağaç, üstten aşağıya doğru büyür. Kök düğümüyle başlayarak, veritabanı "ayırma seçim yöntemi" vasıtasıyla her düğümde ayırma durumunu seçerek incelenebilir. Veritabanı daha sonra bölümlere ayrılır ve işlem tekrar tekrar uygulanır. Budama etabında, ağaç-inşası safhasında yapılan ağaç boyutunu kontol altında tutmak için budanır ve gelişmiş budama metotları ağacı tahmin hatalarını en aza indirecek şekilde seçer [89][83][85][101].

4.1.1. Karakteristik seçme ölçüleri

Bir karakteristik seçme ölçüsü, belirli bir veri bölmesi olan D'deki sınıf-etiketlenmiş örnekleri bireysel sınıflara "en iyi" ayıran ayırma kriterlerini seçmek için kullanılan deneye dayalı bir yöntemdir. Eğer D'yi ayırma kriterlerinin sonuçlarına göre küçük bölmelere ayrılırsa, her bir bölme ideal olarak saf olurdu (mesela, belirli bir bölmeye düşen örneklerin hepsi aynı sınıfa ait olacaktı). Kavramsal olarak, "en iyi" ayırma kriteri böyle bir senaryoya en yakın netice verendir. Karakteristik seçme ölçüleri ayırma kuralları olarak da bilinir çünkü belirli bir düğümdeki örneklerin nasıl ayırılacağına karar verir [74][82].

Karakteristik seçme ölçüleri belirli eğitim örneklerinde herbir karakteristiği tanımlayan bir sıralama temin eder. Ölçmek için en iyi sonuca sahip karakteristik belirli örneklerde ayırma karakteristiği olarak seçilir. Eğer ayırma karakteristiği devamlı-değerli veya eğer ikili ağaçlarla sınırlandırılmışsa, sırasıyla, ya ayırma noktası veya ayırma altkümesi ayırma kriterinin bir parçası olarak da saptanmalıdır. D bölmesi için oluşturulan ağaç düğümü ayırma kriteriyle etiketlenir, dallar kriterin herbir sonucu için yetiştirilir ve örnekler uygun olarak bölümlere ayrılır. Bu bölüm üç popüler karakteristik seçme ölçülerini tanımlamaktadır – bilgi kazanımı, kazanç oranı, ve gini indeksi [89][83][85].

Kullanılan notasyon aşağıdaki gibidir. D veri parçası, sınıf-etiketlenmiş örneklerden oluşan bir eğitim seti olsun. Sınıf etiket özelliği, m farklı sınıfı tanımlayan m farklı değerleri olsun, Ci (i = 1 için, … , m). Ci,D, D'deki, Ci sınıfının örnekler seti olsun. |D| ve |Ci, D| sırasıyla D, Ci, D örneklerin sayısını gösterir [78].

4.1.2. Bilgi kazanımı

ID3 algoritması, bilgi kazanımını karakteristik seçme ölçüsü olarak kullanır. Bu ölçü Claude Shannon'un, mesajların değerini veya "bilgi içeriğini" araştıran bilgi teorisi üzerine öncülük eden eserine dayanır. N düğümü D bölmesindeki örnekleri temsil etsin. En yüksek bilgi kazanımı olan karakteristik N düğümü için ayırma karakteristiği olarak seçilir. Bu karakteristik sonuçta ortaya çıkan bölmelerdeki örnekleri sınıflandırmada gerekli olan bilgiyi en aza indirir ve bu bölmelerde en az rastgelelik veya "saf olmama" yı yansıtır. Böyle bir yaklaşım belirli bir örneği sınıflandırmak için gerekli olan beklenilen test sayılarını en aza indirir ve bir basit (fakat en kolayı zorunlu kılmadan) ağacı bulma garantisini verir. D'deki bir örneği sınıflandırmak için gerekli tahmin edilen bilgi aşağıda verilir [95][75][85]

$$Bilgi(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{4.1}$$

D, rastgele seçilmiş bir örneğin Ci sınıfına ait olduğu yerde pi bir olasılıktır ve |Ci,D|/|D| ile tahmin edilir. Log fonsiyonu 2 tabanını kulanılır, çünkü bilgi bitler şeklinde kodlanmaktadır. Bilgi(D) sadece D deki bir örneğin sınıf etiketini tesbit etmek için gereken ortalama miktardaki bilgidir. Sahip olunan bilgi, yalnız her sınıf örneğinin oranlarına dayanmaktadır. Bilgi(D) D'nin enropisi olarak da bilinir [89][95][75].

D'de bulunan örnekler, {a1, a2, … , av} gibi farklı değerlere sahip olan A karakteristiği üzerinden bölünmek istendiği farzedildiğinde, eğer A ayrık değerliyse, bu değerler doğrudan A üzerindeki bir denemenin v neticelerine karşılık gelir. A karaktestiği, D' yi v altkümelerine {D1, D2, … , Dv }ayırmak için kullanılabilir. Dj, A nın sonuçu olan aj D'deki o örnekleri içerir. Bu bölmeler N düğümünden büyüyen dallara karşılık gelecektir. İdeal olarak, bu bölümlemenin örneklerin kesin bir sınıflandırılmasını meydana getirmesi istenir. Yani, herbir bölme saf olmalıdır. Ancak, bölmelerin saf olmayacağı oldukça muhtemeldir (mesela, bir bölmenin sadece bir sınıftan daha ziyade farklı sınıflardan bir örnekler yığını içerebilir). Kesin bir sınıflandırmaya ulaşmak için daha ne kadar bilgiye (bölümlemeden sonra) ihtiyaç duyacağı şu şekilde ölçülmüştür [89][95][75].

$$Bilgi_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Bilgi(D_j) \tag{4.2}$$

|Dj |/|D| terimi j. bölmesinin ağırlığı gibi davranır. BilgiA(D) A tarafından bölümlenmeye dayanarak D den bir örneği sınıflandırmak için gereken beklendik bilgidir. Gerekli olan beklendik bilgi ne kadar küçük olursa, bölmelerin saflığı daha büyük olur. Bilgi kazancı orjinal bilgi gereksinimiyle (mesela, sadece sınıfların oranına dayalı) yeni gereksinimler (mesela, A bölümlemeden sonra elde edilen) arasındaki fark olarak tanımlanmıştır. Yani,

$$Kazanç(A) = Bilgi(D) - Bilgi_A(D) \tag{4.3}$$

Başka bir deyişle, Kazanç(A), A daki dallanmayla ne kadar kazanacağımızı bildirir. A'nın değerinin bilinmesinden kaynaklan bilgi gereksinimindeki beklenilen azalmadır. En yüksek bilgi kazancı olan A karakteristiği, (Kazanç(A)), N düğümünde ayırma karakteristiği olarak seçilmiştir. Bu A karakteristiğinde "en iyi sınıflandırma" yı yapacak bölümlemeyi yapmak istendiği ayklaşımına denktir. Bundan dolayı örnekleri sınıflandırmayı bitirmek için hala gereken bilgi miktarı en azdır [89][95][75][85].

4.1.3. Kazanç oranı

Bilgi kazanım ölçüsü, çok neticeli testlere karşı önyargılıdır. Yani, çok sayıda değerleri olan karakteristikleri seçmek tercih edilir. ID3'nin bir sonraki versiyonu olan C4.5, bu önyargıyı aşmaya çalışan kazanç oranı olarak bilinen bilgi kazancına bir ilave kullanır. Bilgi(D) ile benzer olarak tanımlanmış "ayırım bilgi" değerini kullanarak bilgi kazancına bir tür normalleşme uygular [91][92][95][83].

$$Bölmebilgisi_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right) \tag{4.4}$$

Bu değer D eğitim veri takımının A karakteristiği üstündeki bir testin v sonuçlarına karşılık gelen v bölmelerine ayrılmasıyla meydan gelen potensiyel bilgiyi temsil eder. Herbir sonuç için, D deki toplam örnek adetlerine nazaran o sonuca sahip olan örnek adetlerini göz önünde bulundurur. Aynı bölümlemeye dayanarak elde edilen sınıflandırmayla ilgili bilgiyi ölçen bilgi kazancından farklılık göstermektedir. Kazanç oranı şu şekilde tanımlanmıştır [91][92][85].

$$KazançOranı(A) = \frac{Kazanç(A)}{Bölmebilgisi(A)} \tag{4.5}$$

Maksimum kazanç oranına sahip olan karakteristik, ayırma karakteristiği olarak seçilmiştir. Ancak, ayırma bilgisi 0'a yaklaşınca, oran dengesiz olur. Bunun önüne

geçmek için bir kısıtlama ilave edilmiştir; seçilen testin bilgi kazancı, en az, incelenen toplam testlerin ortalama kazancı kadar büyük olmalıdır [89].

Gini indeksi

Gini indeksi CART algoritmasında kullanılır. Gini indeksi, bir eğitim seti veya veri seti parçası olan D'nin safsızlığını ölçer [82][86][92].

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2 \tag{4.6}$$

D'deki bir örneğin Ci sınıfına ait olduğu yerde pi bir olasılıktır ve |Ci,D|/|D| ile hesaplanır. Toplam, m sınıfları üzerinden hesaplanır. Gini indeksi, herbir karakteristik için bir ikili ayırımını dikkate alır. İlk olarak A'nın, D veri setinde, ayrık değerli, v farklı değerlere $\{a_1, a_2, \ldots, a_v\}$ sahip olan bir karakteristik olduğu göz önünde bulundurulur. A üzerinde en iyi ikili ayırımını saptamak için, A'nın bilinen değerlerini kullanarak oluşturulan mümkün olan altkümelerin hepsi incelenir [82][86][92].

Her altküme, SA, A karakteristiğinin bir şekli olan "A 2 SA?" için bir ikili test olarak kabul edilebilir. Bir test göz önünde tutulursa, bu test, örneğin eğer A'nın değeri SA'da listelenen değerler içindeyse tatmin edicidir. Eğer A mümkün v değerlere sahipse, o halde mümkün 2^v altküme vardır. Örneğin, eğer gelir karakteristiğinin üç mümkün değeri varsa, yani {düşük,orta, yüksek}, o zaman mümkün altkümeler { düşük,orta, yüksek}, {düşük,orta }, { düşük, yüksek }, { orta, yüksek}, { düşük, { orta }, { yüksek }, and {} dir. Güç kümesi, {düşük,orta, yüksek} ve boş küme göz önünde bulundurulmaz, çünkü, bunlar kavramsal olarak, bir ayrımı temsil etmezler. Bu yüzden, A üzerindeki ikili ayrımı baz alınarak D verisinin iki bölmelerini oluşturmak için 2^v-2 mümkün yol vardır. Bir ikili ayrımı göz önüne aldığımızda, herbir sonuçta ortaya çıkan bölmenin safsızlık ağırlıklı toplamını hesaplarız. Örneğin, eğer A bir ikili ayrımı D'yi D1 ve D2 olarak bölümlerse, belirli bölümlendirilen D'nin gini indeksi,

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \quad (4.7)$$

Herbir karakteristik için, mümkün olan herbir ikili ayrımı göz önünde tutulur. Ayrık-değerli karakteristik için, o karakteristiğe ait en az gini indeksini veren altküme, onun ayrılan altkümesi olarak seçilir. Ayrık veya sürekli değerli olan A karakteristiği üzerindeki bir ikili ayrımından kaynaklanan safsızlıktaki azalma şöyledir [89][82][85][86].

$$\Delta Gini(A) = Gini(D) - Gini_A(D) \quad (4.8)$$

4.1.4. Ağaç budama

Sıklıkla, eğer düğüme ulaşan eğitim örneklerinin adedi eğitim setinin belirli bir yüzdesinden -örneğin, safsızlığa veya hataya bakılmaksızın yüzde 5- daha az ise düğüm daha fazla ayrılmaz. Çok az örneğe dayanan herhangi bir karar, değişikliğe ve böylece genelleştirme hatasına sebebiyet verir. Ağaç oluşturmanın bitme aşamasından önce durdurulmasına önce-budama (prepruning) denir. Pratikte önce-budamadan daha iyi çalışan basit ağaçlar elde etmenin diğer bir yolu, sonra-budamadır (postpruning). Daha önce ağacın hırsla büyüdüğü belirtilmişti; herbir adımda bir karar verilir, yani bir karar düğümünü meydana getirilir ve ileriye doğru devam eder, asla geri-gelme ve başka bir alternatif test edilmez. Tek istisna, gereksiz alt-ağaçları bulmaya ve budamaya çalışılan sonra-budamadır [86][89][102].

Sonra-budamada, ağacı, yaprakları saf olana kadar tamamiyle büyütülür ve eğitim hatası yoktur. Daha sonra ezberlemeye sebep olan alt-ağaçları ve budama kümesi bulunur ve budanır. İlk etiketlenmiş kümeden, eğitim süresinde kullanılmayan budama kümesi atanır. Herbir alt-ağaç için, bir yaprak düğümüyle değiştirilir [103][104].

4.2. Kural Tabanlı Sınıflandırıcılar

Karar ağaçları, kökten, ağaçtaki bir yaprağa kadar herbir yol için ayrı bir kural oluşturarak bir kural tabanına dönüştürülür [105]. Ancak, kurallar, çeşitli kural-tabanlı algoritmalar kullanarak doğrudan eğitim setlerinden de elde edilebilir. Amaç, eğitim verileriyle uyumlu en küçük kural tabanını oluşturmaktır. Öğrenilen kuralların çokluğu, genellikle, algoritmanın veri seti ile ilgili hipotezleri öğrenme, keşfetme yerine, eğitim setini ezberlediğinin bir işaretidir. Böl ve fethet algoritması eğitim örneklerinin bir parçasını açıklayan kuralları arama işlemini gerçekleştirir; veri setinde hiçbir örnek kalmayıncaya kadar, veri setindeki örnekler bölünür ve üretilen kurallar ile kapsanır [93][95][85][92].

Kural-tabanlı sınıflandırıcılar, sınıflandırma işlemi için EĞER- İSE yapısındaki kural setini kullanır. Bir EĞER-İSE kuralı şu şekilde açıklanabilir: Bir kuralın "EĞER" parçası (veya sol tarafı) kuralın önkoşulu olarak bilinir. Bir kuralın "İSE" parçası (veya sağ tarafı) kuralın neticesidir. Kural önkoşuluda, koşul, mantiki olarak bir veya daha fazla karakteristik testlerinden meydan gelir. Kuralın neticesi bir sınıf tahminini kapsar. Eğer, bir kural önkoşulundaki şart (yani, karakteristik testlerinin hepsi) belirli bir örnekte geçerli olursa, kural ön koşulunun sağlandığı ve kuralın örneği kapsadığı kabul edilir. Bir kural K, kendi kapsamı ve doğruluğuyla değerlendirilir. Sınıf etiketli veri seti, D'den verilen bir örneğin X olduğu varsayıldığında, $n_{kapsananlar}$ K'nin kapsadığı örnekler adedi, $n_{doğru}$ K'nin doğru olarak etiketlendirdiği örnek sayısı ve |D|, D'de bulunan örneklerin adedi olsun. K'nın kapsamı ve doğruluğu şu şekilde tanımlanır [95][86].

$$kapsama\,(K) = \frac{n_{kapsananlar}}{|D|} \tag{4.9}$$

$$doğruluk\,(K) = \frac{n_{doğru}}{n_{kapsananlar}} \tag{4.10}$$

Yani, bir kuralın kapsamı kuralın kapladığı örneklerin yüzdesidir. Kuralın doğruluğu için, kapsadığı örneklere bakılır ve kuralın, bunların ne kadarını doğru olarak sınıflandırabildiğini belirlenir. Kural tabanlı sınıflandırma, belirli bir örneğin sınıf etiketini tahmin etmek için şu şekilde kullanılır. Eğer kural, X'e uygunsa, kural tetiklenmiş demektir. Eğer K, uygun olan tek kuralsa, o zaman kural X için sınıf tahminini geri getirerek işlevini yerine getirir (ateşleme). Bu noktada önemli bir kusus söz konusudur; tetikleme, her zaman ateşleme anlamına gelmemektedir. Eğer birden fazla kural bir örnek için tetikleniyorsa, potansiyel bir problem söz konusudur [82][85][86].

Eğer, Kural tabanındaki kuralların herbiri, X için farklı bir sınıf belirlerse, hiçbir kural X'e uygun değilse veya birden fazla kural X için tetiklendiğinde, X'in hangi sınıfa ait olduğunu tahmin etmek için ortaya çıkacak kargaşayı çözmek içinbir takım stratijilere ihtiyaç duyulur. Bu konuda geliştirilmiş iki teknik; boyut sıralaması ve kural sıralamasıdır [86][82][86].

Boyut sıralama planında, en yüksek önceliği "en sağlam" koşula sahip tetiklenmiş kurala verilir. Sağlamlık, kural şartının boyutuyla ölçülür. Yani, en fazla karakteristik testlerine sahip kural ateşlenir (kullanılır). Kural sıralama planında, kurala peşinen öncelik tanınır. Sıralama, sınıf-tabanlı veya kural-tabanlı olabilir. Sınıf-tabanlı sıralamayla, sınıflar azalan yaygınlık sıralaması gibi, azalan "önem" sıralamasıyla türlerine göre ayrılır. Yani, en çok yaygın (veya en sık görülen) sınıfın bütün kuralları sonra gelir. Alternatif olarak, herbir sınıfın yanlış hesaplama masrafına dayanarak ayrılabilir [93][95][85].

Her sınıf içinde, kurallar sıralı değildir – olmamalı çünkü hepsi aynı sınıfı tahmin eder (ve sınıf anlaşmazlığı olamaz). Kural-tabanlı sıralamayla, kurallar doğruluk, kapsam veya boyut (kural önceliğindeki karakteristik testleri adedince), veya alan uzmanlarından tavsiyeye dayanarak bazı kural kalite ölçülerine göre bir uzun öncelik listesi içinde tertiplenmiştir. Kural sıralaması kullanıldığı zaman, kural seti, karar listesi olarak bilinir. Kural sıralamayla, en eski listede görünen ateşleme kuralının en yüksek önceliği vardır, ve böylece kendi sınıf tahminini ateşler. X'e uyan herhangi

diğer bir kural önemsenmez. Kural-tabanlı sınıflandırma sistemlerinin çoğu bir sınıf-tabanlı sıralama stratejisi kullanır. Birinci stratejide, kuralların geneli sırasızdır. Bir örneği sınıflandırırken herhangi bir sıralamanın uygulanabildiğine dikkat edilmeli. Yani, bir ayrışım (mantıki VEYA) herbir kuralın arasında olduğunu belirtir. Her kural tek başına bir bilgi parçasını temsil eder. Kural anlaşmazlıklarını önlemek için belirtilen sıralamada uygulanması kural sıralama (karar listesi) planına aykırıdır. Karar listesindeki her kural listede kendinden önde gelen kuralların eksikliğini belirtir. Bundan dolayı, bir karar listesindeki kurallar yorumlamak çok zordur [74][75][83].

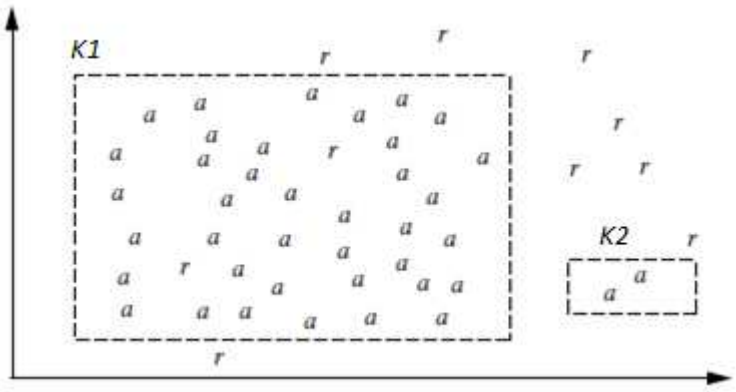
Kural anlaşmazlıklarının yanında, X'e uyan herhangi bir kural olmama durumunda nasıl bir strateji izlenmeli? Bu duruma, eğitim setine dayanarak öntanımlı sınıfı belirlemek için öntanımlı bir kural düzenlenebilir. Bu, çoğunlukta olan sınıf veya herhangi bir kuralla ele alınmayan örneklerin çoğunluk sınıfı olabilir [74][75][83].

4.2.1. Kural kalite ölçüleri

Algoritmaların doğruluk oranları, ilk olarak ayırt edici bir seçenek gibi görünür, fakat aşağıdaki örnekte açıklandığı gibi, doğruluğa dayanarak iki kural arasında seçim yapıldığında, aşılması gereken bazı problemlerin ortaya çıktığı görülmektedir [89][92][82]. İki kuralın yapısının, Şekil 4.2'de belirtildiği gibi olduğu varsayılsın. Her iki kuralda, sınıf seçimi için "a"'ya karar vermektedir. K1 kuralı 40 örnekten 38'ini kapsarken, K2 kuralı sadece 2 örneği kapsamaktadır. Bunun yanında, K1 örneğinin doğruluk oranı %95 ikin, K2 kuralının oranı %100'dür. Yani, K2 , K1'den daha yüksek bir doğruluk oranına sahiptir. Fakat, dar kapsama kabiliyetinden dolayı iyi bir kural değildir [85].

Yukarıdaki örnekte, doğruluğun, tek başına güvenilir bir kural kalite tahmini olmadığı görülür. Bunu yanında, kapsama ölçüsüde tek başına bir kriter değildir. Bu nedenle, kural kalitesini değerlendirmek için, doğruluk ve kapsamın kavramlarını birleştiren yöntemler geliştirilmiştir. Bunlardan bazıları, entropi, bilgi kazancı ve kapsamı dikkate alan istatistiki test yöntemleridir [86][87].

Örnek olarak, c sınıfı için kuralların üretildiği farzedilsin. Aktif kural K: EĞER koşul İSE sınıf=c dir. Belirli bir karakteristik testinin mantıksal olarak bitmesine şartlandırırmanın iyi bir kuralla sonuçlanıp sonuçlanmayacağı görmek istenir.. Yeni koşulu koşul0,K0'ın olduğu yerde: EĞER koşul0 SONRA sınıf=c potansiyel yeni kuraldır. Başka bir deyişle, K0'ın K'dan daha iyi olduğu görmek istenir [86][87] .



Şekil 4.2. Kuralın örnek kapsama yapısı

Entropi kavramı, karar ağaçlarında, karakteristik seçimi için kullanılan bilgi kazanç ölçüsü konusunda açıklanmıştı. D veri setindeki, bir örneği sınıflandırmak için gerekli olan beklenilen bilgi olarak da bilinir. Burada D, koşul0 tarafından kapsanan örnekler seti ve pi, D'deki Ci sınıfının olasılığıdır. Entropi ne kadar düşük olursa, koşul0 o kadar iyi olur. Entropi, tek bir sınıftaki çok sayıdaki örneği ve diğer sınıfların az sayıdaki örneklerinin kapsandığı koşulları tercih eder. Diğer bir ölçü, bilgi kazanımına dayalıdır ve FOIL (First Order Inductive Learner) metodunda önerilmiştir [16][83].

Bilgisayar öğrenmeside, kural üretilmeye çalışılan sınıfa ait örneklere, pozitif örnekler denirken, geri kalan örnekler negatif'dir. Pos(neg), K kuralı tarafından kapsanan pozitif (negatif) örneklerin sayısı olsun. Pos0 (neg0), K0 tarafından kapsanan pozitif (negatif) örneklerin sayısı olsun. FOIL, genişletilmiş koşulla kazanılan bilgiyi şu şekilde hesaplar[15][16].

$$FOIL\_Gain = pos' \times \left(log_2 \frac{pos'}{pos' + neg'} - log_2 \frac{pos}{pos + neg} \right) \quad (4.11)$$

Bu yaklaşım, yüksek doğruluğu olan ve bir çok pozitif örneği kapsayan kuralları tercih eder.

Ayrıca, bir istatistiki önem testi, eğer bir kuralın görünen etkisinin şansa değil, karakteristik değerleri ve sınıfları arasındaki gerçek bir ilişkinin olduğunu saptamada da kullanabiliriz. Test bir kural tarafından kapsanan örnekler sınıfları arasındaki kuralın rastgele tahminler yaparak sonuçlanmsıyla gözlemlenen dağılımı karşılaştırır. Bu iki dağılım arasındaki gözlemlenen farklılıkların şansa bağlanıp bağlanamayacağını ölçmek istenir. İhtimal oran istatistiği kullanılır,

$$ihtimal\_Orani = 2 \sum_{i=1}^{m} f_i log\left(\frac{f_i}{e_i}\right) \quad (4.12)$$

Burada m, sınıf adedidir. Kurala uyan örnekler için, f_i, örnekler arasında herbir i sınıfının gözlemlenen frekansıdır. e_i, kural rastgele tahminler yaptığında, herbir i sınıfının olması beklenen frekansıdır. İstatistik, m-1 serbeslik dereceli, x^2 dağılımına sahiptir. İhtimal oranı yüksek olursa, büyük olasılıkla bu kuralın bir "rastgele tahminci"ye kıyasla yaptığı doğru tahminlerin sayısında önemli farklılıklar vardır. Yani, kuralın performansı şanstan dolayı değildir. Oran, önemsiz kapsamı olan kuralları tesbit etmeye yardımcı olur. CN2 algoritması entropiyi, ihtimallik oran testiyle birlikte kullanırken, RIPPER algoritması, FOIL bilgi kazanımını kullanır [14][15][16].

4.2.2. Kural budama

Bir Kural Öğren (Learn One Rule) algortması, kuralları değerlendirirken bir test seti kullanmaz. Kural kalitesinin değerlendirmeleri yukarıda bahsedildiği gibi orjinal eğitim verilerinden örneklerle yapılır. Böyle bir değerlendirme iyimserdir çünkü kurallar ihtimalen verileri ezberlemiş olacaktır. Yani, kurallar eğitim verilerinde iyi, fakat test verilerde az performans sağlayabilir. Bunu telafi etmek için, kurallar budanır. Eğer K'nin budanmış versiyonu, bağımsız örnekler setinde ölçüldüğü gibi

yüksek kaliteye sahipse, bu kural budamak seçilir. Karar ağacı budaması gibi, bu setten budama seti olarak bahsedilir. Bir önceki bölümde açıklanan karamsar budama yaklaşımı gibi çeşitli budama stratejileri kullanılabilir. FOIL basit fakat etkili bir yöntem kullanır. Verilen bir kural K için

$$FOIL\_Budama(R) = \frac{pos - neg}{pos + neg} \tag{4.13}$$

burada, pos ve neg sırasıyla K'nın kapsadığı pozitif ve negatif örnek adetleridir. Bu değer K'nın budama testindeki doğruluğuyla birlikte artacaktır. Bu yüzden, eğer FOIL budama değeri K'nın budanmış versiyonundan yüksek ise, o zaman K' budanır [15].

BÖLÜM 5. IREM (INDUCTIVE RULE EXTRACTION METHOD) ALGORİTMASI

5.1. Giriş

Bu bölümde yeni geliştirilen bilgisayar öğrenme algoritması olan IREM (Inductive Rule Extraction Method) sunulacaktır. Algoritmanın özellikleri ayrıntılı olarak verilerek ve örnek bir veri setin üzerinden kural üretme prosedürü açıklanacaktır. Ayrıca, gerçek veri setleri yardımı ile algoritmanın performansı aktarılacaktır.

5.2. Algoritmanın Tanımı

IREM algoritması, bilgisayar öğrenmesi alanında, danışmanlı öğrenme algoritmaları kategorisinde, kapsama yöntemini kullanarak kural çıkartan yöntemler arasında yer almaktadır. Algoritma, belirli bir disiplin tarafından üretilen özel bir örnek setinden EĞER-İSE (IF-THEN) yapısında genel kurallar üretilir. Kural üretme prosedüründe, bilgi değeri yüksek veriler seçilerek genel kurallar çıkarılması hedeflenmektedir. Bu bağlamda karakteristik-değer ikililerinin olasılık dağılımları ve entropileri kullanılarak yeni bir maliyet fonksiyonu geliştirilmiştir.

Mantıksal/sembolik kategorideki bilgisayar öğrenme algoritmalarından karar ağaçları ve kural tabanlı metodlar arasından bir kısmı (CLS ve RULES ailesi algoritmaları gibi) veri setindeki verilerin bilgi değerleri ile ilgilenmeden kural üretirken, bir kısmı (ID3, C4.5 ve REX ailesi algoritmaları gibi) ise karakteristik-değer ikililerinin bilgi değerlerini dikkate alarak kural üretme işlemlerini yönetmektedirler.

Örnek olarak, RULES-3 algoritması verilen örnek setindeki örnekleri sırayla ele alır. Her örnekteki karakteristik değerlerinin oluşturulan kombinasyonlarından herhangi

birisi bütün örnek setinde sadece bir sınıfta geçiyorsa bu değer kural haline getirilir. Aksi halde farklı kombinasyonlara bakılır. Bu işlem karakteristik sayısı kadar kombinasyona ulaşılıncaya kadar tekrar edilir. Ancak RULES ailesi algoritmalarında karakteristiklere ait bilgi değerleri göz önüne alınmadığından çıkarılan kural sayısı fazla olabilmekte ve kurallarda bilgi değeri küçük olan karakteristiklerde yer alabilmektedir. Karar ağaçları üreten CLS ailesi algoritmalarında da bilgi değerleri göz önünde bulundurmadan kural üretilmektedir. Bundan dolayı RULE-3 algoritmasında oluşan sıkıntıları paylaşmaktadır.

Bununla beraber, veri setindeki herbir değerlerin ve karakteristiğin entropisini hesaplayarak, bilgi kazancı en yüksek karakteristik değer ikililerine öncelik verilen algoritmaların daha genel kurallar ürettiği görülmektedir. ID3, C4.5 ve REX algoritmaları örnek olarak verilebilir.

REX-1 algoritması ilk önce örnek setinde bulunan karakteristik ve bu karakteristiklere ait değerlerin entropilerini hesaplayarak, düzensizliği az olan karakteristiklere öncelik vererek kural çıkarma işlemini gerçekleştirir. Entropisi düşük olan karakteristiklerin bilgi kazancı yüksektir. Bundan dolayı, örnek setinde bilgi değeri en büyük olan karakteristiklere öncelik verilir.

Geliştirilen IREM algoritması da bilgi değeri hesabı gerçekleştirerek kural üretilmesini sağlayan bir algoritmadır. IREM'in, diğer bilgisayar öğrenme algoritmalarından farkı, veri setindeki karakteristik-değer ikililerinin bilgi değerlerini sınıf bazında hesaplanmasına imkan sağlayan yeni bir maliyet fonksiyonu kullanmasıdır. Maliyeti az olan karakteristik-değer ikilisinin bilgi değeri yüksek olduğu kabul edilmektedir. Böylece, oluşturulan kural tabanı ile, hata oranın düşürülmesi hedeflenmiştir.

IREM algoritmasının işleyişine geçmeden önce, algoritmanın temelini oluşturan maliyet fonksiyonu ve maliyet fonksiyonunun kullandığı entropi kavramı üzerinde durulacaktır.

5.2.1. Entropi

Entropi, en kaba tanımıyla bir sistemin düzensizliğinin ölçüsüdür. Birçok disiplin içinde ayrı ayrı entropi fonksiyonları kullanılır. Örneğin termodinamik entropi, topolojik entropi gibi. Sistemin düzensizliği arttıkça artan herhangi bir fonksiyon entropi fonksiyonu olabilir [106].

Bilgi ölçümü, olasılık ile ters orantılıdır. Örneğin, bir x çıktısı için olasılık p(x) ise, aynı çıktı için bilgi ölçümü (I(x)) denklem (5.1) gibi gösterilir

$$I(x) = f\left(\frac{1}{p(x)}\right) \tag{5.1}$$

x1 ve x2 gibi bağımsız iki çıktının olasılığı, ikisinin ayrı ayrı olasılıklarının çarpımına eşittir.

$p(x_1 \text{ ve } x_2)=p(x_1)\ p(x_2)$

(5.2) eşitliğinin her iki tarafının 1'e bölünmesi durumunda eşitlik değişmeyecektir.

$$\frac{1}{p(x_1\ ve\ x_2)} = \frac{1}{p(x_1)}\frac{1}{p(x_2)} \tag{5.3}$$

Denklem (5.3)'ün her iki tarafının logaritması alındığında (5.4) nolu denklem elde edilir.

$$\log\frac{1}{p(x_1\ ve\ x_2)} = \log\frac{1}{p(x_1)} + \log\frac{1}{p(x_2)} \tag{5.4}$$

Ayrıca x1 ve x2 gibi iki çıktının bilgi ölçümü, bu çıktıların ayrı ayrı bilgi ölçümleri toplamına eşittir. Bu işlem denklem (5.5) ile gösterilebilir.

$I(x_1 \text{ VE } x_2)=I(x_1)+I(x_2)$ (5.5)

$$I(x_1) = \log\left(\frac{1}{p(x_1)}\right) \tag{5.6}$$

(5.6) nolu denklem genelleştirildiğinde, bir x çıktısı için bilgi ölçümünü veren denklem, (5.7)'deki denklem elde edilir.

$$I(x) = f\left(\frac{1}{p(x)}\right) = \log\left(\frac{1}{p(x)}\right) \tag{5.7}$$

Bir olayın çıktılarındaki ortalama bilgi ölçümü, bir olayın entropisi olup olayın mümkün olan tüm çıktılarının bilgi ölçümlerinin ağırlıklı ortalaması olarak hesaplanır. Burada kullanılan ağrılıklar çıktıların olasılıklarıdır.

Sadece iki çıktısı olabilen bir X olayı için, bu olayın olasılıkları p ve (1-p)'dir. Yukarıda verilen tanıma göre bu olayın entropisi E(X), denklem (5.8) ile hesaplanabilir.

$$E(X) = p\log_2\frac{1}{p} + (1-p)\log_2\frac{1}{1-p} \tag{5.8}$$

Burada I(p)=Log(1/p) yazıldığı takdirde (5.9) nolu denklem elde edilir.

$$E(X) = p \cdot I(p) + (1-p) \cdot I(1-p) \tag{5.9}$$

Buna göre n adet çıktısı olan bir X olayının entropisi (5.10), (5.11) ve (5.12) yardımıyla hesaplanabilir.

$$E(X) = \sum_{i=1}^{n} p_i I_i \tag{5.10}$$

$$E(X) = \sum_{i=1}^{n} p_i \log\left(\frac{1}{p_i}\right) \tag{5.11}$$

$$E(X) = -\sum_{i=1}^{n} p_i \log(p_i) \tag{5.12}$$

Burada logaritma için istenilen herhangi bir taban kullanılabilir. İki tabanının kullanılması daha yaygındır. Ancak bu durumda bilgi ölçümü ve entropi için birim olarak bit (binary digit) kullanılabilir.

5.2.2. Maliyet fonksiyonu

IREM alogoritmasında, kural üretme prosedürüne alınacak karakteristik-değer ikililerinin seçim işlemi, aktif kaydın sınıfına bağlı olarak, en düşük maliyet değerine sahip olanlardan başlar. Çünkü, maliyeti düştükçe, bilgi değerinin yükseldiği düşünülmektedir. Bu bağlamda, IREM algoritmasında kullanılan maliyet hesabının yapılabilmesi için aşağıdaki yöntem geliştirilmiştir.

$P(V_C)$: veri setindeki her değerin sınıf bazındaki koşullu olasılıkları

$|V_C|$: C sınıfında olan V değerlerinin sayısı

$|V|$: veri setindeki tüm V değerlerinin sayısı

$T_{V,C}$: V değerinin C sınıfındaki olasılığının tümleyeni

E(V) : veri setindeki değerlerin entropileri

E(X) : veri setini oluşturan karakteristiklerin entropileri

M(V|C) : veri setindeki değerlerin sınıf bazındaki maliyetleri

$X=\{x_1, x_2, x_3,\ldots, x_y\}$ y, veri setindeki karakteristik sayısı

$V=\{v_1, v_2, v_3,\ldots, v_n\}$ n, veri setindeki tüm değerlerin sayısı

$C=\{ c_1, c_2, c_3,\ldots, c_m \}$ m, sınıf sayısı

olmak üzere, her değerin sınıf bazındaki maliyeti M(V|C), şu şekilde hesaplanır.

$$P(V_C) = |V_C| / |V| \tag{5.13}$$

$$T_{V,C} = 1 - P(V_C) \tag{5.14}$$

$$M(V|C) = \begin{cases} 0, & P(V_c) = 1 \\ -1, & P(V_c) = 0 \\ T_{V,C} * E(V) + T_{V,C} * E(X), & 0 < P(V_c) < 1 \end{cases} \tag{5.15}$$

Entropi tanımlamasında ifade edildiği gibi, entropi, bir sistemdeki belirsizliğin derecesidir. Entropi, herbir değerin sınıf bazındaki maliyetlerinin hesaplanması için tek başına yeterli gelmemektedir. Bundan dolayı, entropi ile beraber, değerlerin sınıf bazındaki olasılık dağılımlarından istifade edilmiştir.

Olasılık dağılımlarında, bir değerin herhangi bir sınıftaki olasılığı 1 ve ya 0 ise aktif değerin bu sınıfta belirsizliği yoktur. Yani kesin bir kanıya varılabilir. Eğer, sözü edilen olasılık 1 ise üzerinde çalışılan değerin, olasılığın 1 olduğu sınıftaki maliyeti 0'dır. Kesinlik olduğundan bu değer ile doğrudan kural üretilir. Olasılık 0 olduğunda ise, aktif değer ile olasılığın 0 olduğu sınıfta bir kural üretmenin mümkün olmadığı kabul edilir. Bu durum, maliyet fonksiyonunda -1 olarak gösterilmiştir (IREM algoritmasının kodlanmasında -1 değerinden istifade edilmektedir).

Olasılıkların 1 veya 0 olması kesinliği ifade etmektedir, yani herhangi bir düzensizlik, belirsizlik yoktur. Fakat, bir değerin herhangi bir sınıftaki olasılığı 0 ile 1 arasında ise bilgi değerlerinin ölçülmesi için sınıf bazındaki maliyetlerin hesaplanması önem kazanmaktadır. Bu çerçevede maliyet hesaplamasında, karakteristik ve değerlerin entropileri ile değerlerin sınıf bazındaki koşullu olasılıklarının tümleyenleri kullanılmıştır.

Karakteristik ve değerlerin entropilerinde sınıf bilgisi bulunmamaktadır. Fakat, veri setindeki değerlerin maliyetlerinin hesaplanması sınıf bazlı olarak gerçekleşmektedir. Bundan dolayı, değerlerin sınıf bazındaki koşullu olasılıklarının tümleyenleri, entropilerin katsayıları olarak kullanılarak sınıf bilgisi ilave edilmiştir.

Değerlerin sınıf bazındaki koşullu olasılıklarının tümleyenlerinin kullanılma sebebi şudur; değerlerin olasılıkları belirli bir oranda kesinliği ifade etmektedir. Düzensizliği ortaya çıkartan ise tümleyen kısım olduğu düşünülmüştür. Bu suretle, entropilerde katsayı olarak kullanılarak, sınıf bilgisi barındıran maliyet hesaplamaları gerçekleştirilmiştir.

5.3. IREM Algoritmasının İşleyişi

Geliştirilen IREM algoritmasının işleyişi aşağıda adım adım açıklanmıştır. Algoritmada geçen sınıf bazında olasılık dağılımları, entropi ve maliyet fonksiyonu gibi kavramlar örnek uygulama vasıtasıyla ayrıntılı olarak anlatılacaktır.

(N: karakteristik sayısı, n : kombinasyon sayısı olmak üzere)

Adım-1. Verilen örnek setindeki tüm karakteristik değerlerinin, sınıf bazında olasılık dağılımları hesaplanır.

Adım-2. Veri setindeki her karakteristiğin ve değerin entropileri hesaplanır.

Adım-3. Olasılık dağılımları ve entropiler kullanılarak, tüm karakteristik-değer ikililerinin maliyetleri hesaplanır.

Adım-4. Her örnekteki değerlerin tekli (n=1) kombinasyonları ele alınır. n=1 için entropisi sıfır olan değerler tek başlarına kural oluşturabilirler. Bu değerler kural haline getirilir (Bu değerlerden aynı örneklere karşılık gelen kurallardan, en çok örneği kapsayan kural geçerli olarak kabul edilir). Sınıflandırılan örnekler işaretlenir.

Adım-5. Adım-8'e git.

Adım-6. Sınıflandırılmamış örnekler sırasıyla ele alınır. Aktif örneğin sınıfına bağlı olarak, sınıf bazındaki maliyeti en düşük karakteristik değerinden başlamak şartı ile, karakteristik değerlerinden n'li kombinasyonlar oluşturulur.

Adım-7. Her kombinasyon örnek setindeki tüm örneklere uygulanır. n adet kombinasyondan oluşan değerlerden tek bir sınıfa karşılık gelenleri kural haline getirilir. Sınıflandırılan örnekler işaretlenir.

Adım-8. Eğer tüm örnekler sınıflandırılmış ise Adım-11'ye gidilir.

Adım-9. n=n+1 işlemi yapılır.

Adım-10. Eğer n<N ise Adım-6'ye gidilir.

Adım-11. Oluşturulan kural tabanı, aynı örnekleri temsil eden kurallar arasından en genel kurallar seçilirek optimize edilir.

Adım-12. SON

5.4. IREM ile Örnek Uygulama

Yukarıda verilen bilgiler çerçevesinde, mevsimler veri seti örneği kullanılarak IREM'in kural üretme adımları ayrıntılı olarak ele alınacaktır. Mevsimler veri seti, yönetilebilir bir yapısı olmasından ve tutarlı örnekler içermesinden dolayı, bilgisayar öğrenmesi alanında çalışan araştırmacıların, algoritmalarını geliştiriken tercih ettiği veri setleri arasında yer almaktadır.

Tablo 5.1'de verilen örnek seti 11 örnekten (m = 11), üç karakteristik (Hava, Ağaçlar, Sıcaklık) ve dört sınıftan (İlkbahar, Yaz, Sonbahar, Kış) oluşmaktadır. Örnek setinde yer alan karakteristikler ve bunlara ait değerler aşağıda verilmiştir:

Karakteristik	**Değerler**
Hava	Yağmurlu, Karlı, Güneşli
Ağaçlar	Yapraksız, Sarı, Yeşil
Sıcaklık	Normal, Düşük, Yüksek

Tablo 5.1. Mervsimler eğitim seti

Örnek No	**Hava**	**Ağaçlar**	**Sıcaklık**	**Mevsim**
1	Yağmurlu	Yapraksız	Normal	Sonbahar
2	Yağmurlu	Yapraksız	Düşük	Kış
3	Karlı	Yapraksız	Düşük	Kış
4	Güneşli	Yapraksız	Düşük	Kış
5	Yağmurlu	Sarı	Normal	Sonbahar
6	Yağmurlu	Yeşil	Yüksek	Yaz
7	Yağmurlu	Yeşil	Normal	İlkbahar
8	Güneşli	Yeşil	Normal	İlkbahar
9	Güneşli	Yeşil	Yüksek	Yaz
10	Güneşli	Sarı	Normal	Sonbahar
11	Karlı	Yeşil	Düşük	Kış

Adım-1. Verilen örnek setindeki tüm karakteristik değerlerinin, sınıf bazında olasılık dağılımları hesaplanır.

P(yağmurlu $_{sonbahar}$) = |yağmurlu sonbahar | / |yağmurlu|

P(yağmurlu $_{sonbahar}$) = 2/5 = 0,40

Diğer değerlere ait olasılık dağılımlar da benzer şekilde hesaplanır. Tablo 5.2'de tüm değerlerin sınıf bazındaki olasılık dağılımları gösterilmiştir.

Tablo 5.2. Değerin sınıf bazındaki koşullu olasılıkları

Karakteristik	Değer	Sonbahar	Kış	Yaz	İlkbahar
Hava	Yağmurlu	0,40	0,20	0,20	0,20
	Karlı	0,00	1,00	0,00	0,00
	Güneşli	0,25	0,25	0,25	0,25
Ağaçlar	Yapraksız	0,25	0,75	0,00	0,00
	Sarı	1,00	0,00	0,00	0,00
	Yeşil	0,00	0,20	0,40	0,40
Sıcaklık	Normal	0,60	0,00	0,00	0,40
	Düşük	0,00	1,00	0,00	0,00
	Yüksek	0,00	0,00	1,00	0,00

Adım-2. Veri setindeki her karakteristiğin ve değerin entropileri hesaplanır.

Mevsimler tablosunda, Hava karakteristiği, Yağmurlu, Karlı ve Güneşli olmak üzere üç değerden oluşmaktadır.

Hava karakteristiğinin Yağmurlu değeri 5 örnekte geçmektedir. Bu örneklerin ikisi Sonbahar, biri Yaz, biri Kış ve biri de İlkbahar sınıflarına aittir. Bu bilgilere göre {Hava,Yağmurlu} için entropi değeri;

$$E_{Hava,Yağmurlu} = -\frac{2}{5}\log_2\frac{2}{5} - \frac{1}{5}\log_2\frac{1}{5} - \frac{1}{5}\log_2\frac{1}{5} - \frac{1}{5}\log_2\frac{1}{5}$$

$E_{Hava,Yağmurlu} = 1.922 \ \ bit$ olur.

Hava karakteristiğinin Karlı değeri 2 örnekte geçmektedir. Bu örneklerin her ikisi de Kış mevsimine aittir. Buna göre {Hava, Karlı} için entropi değeri;

$$E_{Hava,Karlı} = -\frac{2}{2}\log_2 \frac{2}{2}$$

$E_{Hava,Karlı} = 0 \ \ bit$ olur.

Hava karakteristiğinin Güneşli değeri dört örnekte geçmektedir. Her bir örnek farklı sınıflara aittir. Buna göre {Hava,Güneşli} için entropi değeri;

$$E_{Hava,Güneşli} = -\frac{1}{4}\log_2 \frac{1}{4} - \frac{1}{4}\log_2 \frac{1}{4} - \frac{1}{4}\log_2 \frac{1}{4} - \frac{1}{4}\log_2 \frac{1}{4}$$

$E_{Hava,Güneşli} = 2.0 \ \ bit$ olarak hesaplanır.

Bu bilgilere göre Hava karakteristiğinin entropi değeri aşağıdaki gibi olur:

$$E_{Hava} = \frac{5}{11} x E_{Hava,Yağmurlu} + \frac{2}{11} x E_{Hava,Karlı} + \frac{4}{11} x E_{Hava,Güneşli}$$

$$E_{Hava} = \frac{5}{11} x(1.922) + \frac{2}{11} x(0) + \frac{4}{11} x(2.0)$$

$$E_{Hava} = 1.601 \, bit$$

Yukarıda gösterilen örnek çerçevesinde, herbir karakteristik ve değerin hesaplanmış entropileri Tablo 5.3'te toplu olarak verilmiştir.

Tablo 5.3. Karakteristik ve değerlere ait entropi değerleri

Karakteristik	**Entropi (bit)**	**Değer**	**Entropi (bit)**
Hava	1.601	Yağmurlu	1.922
		Karlı	0
		Güneşli	2.0
Ağaçlar	0.987	Yapraksız	0.189
		Sarı	0
		Yeşil	1.522
Sıcaklık	0.441	Normal	0.971
		Düşük	0
		Yüksek	0

Adım-3. Olasılık dağılımları ve entropiler kullanılarak, tüm karakteristik-değer ikililerinin sınıf bazındaki entropileri hesaplanır.

Örnek olarak, hava karakteristiğinin yağmurlu değeri için sonbahar sınıfındaki entropisini hesaplayalım.

$T\ yağmurlu,_{sonbahar} = 1 - P(yağmurlu_{sonbahar}) = 1 - 0,40 = 0,60$

$M(yağmurlu|sonbahar) = T\ yağmurlu,_{sonbahar} *(E(yağmurlu) + E(sonbahar))$

$M(yağmurlu|sonbahar) = 0,60*1,922 + 0,60*1,601$

$M(yağmurlu|sonbahar) = 2,114\ bit$

Mevsimler tablosundaki herbir değerin sınıf maliyet değerleri Tablo 5.4'te gösterilmişti.

Tablo 5.4. Sınıf bazlı maliyet değeri

Özellik	Değer	Sonbahar	Kış	Yaz	İlkbahar
Hava	Yağmurlu	2,114	2,818	2,818	2,818
	Karlı	-1,000	0,000	-1,000	-1,000
	Güneşli	2,701	2,701	2,701	2,701
Ağaçlar	Yapraksız	0,882	0,294	-1,000	-1,000
	Sarı	0,000	-1,000	-1,000	-1,000
	Yeşil	-1,000	2,007	1,505	1,505
Sıcaklık	Normal	0,565	-1,000	-1,000	0,847
	Düşük	-1,000	0,000	-1,000	-1,000
	Yüksek	-1,000	-1,000	0,000	-1,000

Aşağıdaki tabloda açıkça görülebileceği gibi, veri setindeki her değerin sınıf bazındaki maliyetleri birbirinden farklıdır. Algoritmanın temelini oluşturan bu anlayış çerçevesinde, Adım-4'ten itibaren kural üretme prosedüründe Tablo 5.4'teki sıralamaya bağlı kalınarak karakteristik-değer ikililerinin seçimi gerçekleştirilecektir.

Tablo 5.5. Değerlerin sınıf bazında maliyetlerine göre sıralaması

Sonbahar		Kış		Yaz		İlkbahar	
Değer	Maliyet	Değer	Maliyet	Değer	Maliyet	Değer	Maliyet
Sarı	0,000	Karlı	0,000	Yüksek	0,000	Normal	0,847
Normal	0,565	Düşük	0,000	Yeşil	1,505	Yeşil	1,505
Yapraksız	0,882	Yapraksız	0,294	Güneşli	2,701	Güneşli	2,701
Yağmurlu	2,114	Yeşil	2,007	Yağmurlu	2,818	Yağmurlu	2,818
Güneşli	2,701	Güneşli	2,701	Karlı	-1	Karlı	-1
Karlı	-1	Yağmurlu	2,818	Yapraksız	-1	Yapraksız	-1
Yeşil	-1	Sarı	-1	Sarı	-1	Sarı	-1
Düşük	-1	Normal	-1	Normal	-1	Düşük	-1
Yüksek	-1	Yüksek	-1	Düşük	-1	Yüksek	-1

Adım-4: Tablo 5.5'te maliyeti sıfır olan değerler,

Sıcaklık karakteristiği için; Düşük ve Yüksek

Ağaçlar karakteristiği için; Sarı

Hava karakteristiği için; Karlı

Bu değerler tekli kombinasyonlara göre aşağıdaki kuralları oluştururlar:

Kural-1: EĞER Sıcaklık Düşük İSE Mevsim KIŞ

Bu kural ile 2, 3, 4 ve 11 nolu örnekler sınıflandırılır. Bu örneklerin tümü sadece KIŞ sınıfına aittir.

Kural-2: EĞER Sıcaklık Yüksek İSE Mevsim YAZ

Bu kural ile 6 ve 9 nolu örnekler sınıflandırılır. Bu örnekler sadece YAZ sınıfına aittir.

Kural-3: EĞER Ağaçlar Sarı İSE Mevsim SONBAHAR

Bu kural ile 5 ve 10 nolu örnekler sınıflandırılır. Bu örnekler sadece Sonbahar sınıfına aittir.

Kural-4: EĞER Hava Karlı İSE Mevsim KIŞ

Bu kural ile 3 ve 11 nolu örnekler sınıflandırılır. Ancak 3 ve 11 nolu örnekler daha önce Kural-1 ile sınıflandırıldığı için bu örnekler için tekrar kural oluşturulmaz. Bu kural geçersiz olarak kabul edilir.

Bu işlemlerden sonra Adım-8'e gidilir. Tüm örnekler sınıflandırılmadığından dolayı Adım-9'daki n=n+1 işlemi ile n'nin değeri bir artırılarak iki olur ve Adım-6'ya gidilir.

Adım-6: Sınıflandırılmamış örnekler sırasıyla ele alınır. Aktif örneğin sınıfına bağlı olarak, sınıf bazındaki maliyeti en düşük karakteristik değerinden başlamak şartı ile, karakteristik-değer ikililerinden n'li kombinasyonlar oluşturulur.

Sınıflandırılamayan örnekler olarak 1, 7 ve 8 nolu örnekler bulunmaktadır. Bu örnekler Tablo 5.6'da gösterilmiştir.

Tablo 5.6. Birinci aşama sonunda sınıflandırılamayan örnekler

Örnek No	Hava	Ağaçlar	Sıcaklık	Mevsim
1	Yağmurlu	Yapraksız	Normal	Sonbahar
7	Yağmurlu	Yeşil	Normal	İlkbahar
8	Güneşli	Yeşil	Normal	İlkbahar

6. Adımdaki şarta bağlı olarak, birinci örnekteki değerlerin ikili kombinasyonları oluşturulur.

M(normal|sonbahar) = 0,565

M(yapraksız|sonbahar) = 0,882

M(yağmurlu|sonbahar) = 2,114

Maliyet değerlerine bağlı olarak sırasıyla,

{Normal,Yapraksız}, {Normal, Yağmurlu }, {Yapraksız, Yağmurlu}

ikili kombinasyonlar oluşturulur.

Adım-7:

İkili kombinasyonların ilki olan {Normal,Yapraksız} değerleri için tüm örnek setine bakılır. Bu değerler ile sadece 1 nolu örnek sınıflandırılabilmektedir.

Kural-4 : EĞER Sıcaklık Normal VE Ağaçlar Yapraksız İSE Mevsim SONBAHAR

Sınıflandırılamayan örnekler olarak 7 ve 8 nolu örnekler kalmaktadır. Yedinci örnekteki değerlerin ikili kombinasyonları oluşturulur.

M(normal| ilkbahar) = 0,847

M(yeşil| ilkbahar) = 1,505

M(yağmurlu|ilkbahar) = 2,818

Maliyet değerlerine bağlı olarak sırasıyla,

{Normal,Yeşil}, {Normal,Yağmurlu} ve {Yeşil, Yağmurlu}

ikili kombinasyonlar oluşturulur.

{Normal, Yeşil} değerleri 7 ve 8 nolu örneklerde sadece bir sınıfta geçtiklerinden kural haline getirilir.

Kural-5 : EĞER Sıcaklık Normal VE Ağaçlar Yeşil İSE Mevsim İLKBAHAR

Sınıflandırılacak başka örnek kalmadığından dolayı işlemler tamamlanır. Çıkarılan kurallar Tablo 5.7'de gösterilmiştir.

Tablo 5.7. Mevsimler eğitim seti için üretilen kurallar

Kural No	Kural Tanımı
1	**EĞER** Sıcaklık=Düşük **İSE** Mevsim=KIŞ
2	**EĞER** Sıcaklık=Yüksek **İSE** Mevsim=YAZ
3	**EĞER** Ağaçlar=Sarı **İSE** Mevsim=SONBAHAR
4	**EĞER** Sıcaklık=Normal **VE** Ağaçlar=Yeşil **İSE** Mevsim=İLKBAHAR
5	**EĞER** Sıcaklık=Normal **VE** Ağaçlar=Yapraksız **İSE** Mevsim=SONBAHAR

5.5. Deneyler ve Performans Karşılaştırmaları

Deneyler ve performans karşılaştırmaları için, UCI Bilgisayar Öğrenmesi Veri Havuzunda (The University of California, Irvine Machine Learning Repository) bulunan gerçek veri setleri üzerinde çalışılmıştır. UCI Bilgisayar Öğrenmesi Veri Havuzu, geliştirilen bilgisayar öğrenme algoritmalarının deneysel olarak analiz edilmesini sağlayan, gerçek veri setlerinden oluşan bir koleksiyondur. Veri setlerinin, kural çıkarma ve çıkarılan kuralları test etmek için eğitim ve test setleri bulunmaktadır. İlk kez 1987'de David Aha tarafından oluşturulan arşiv, 2007'de Arthur Asuncion and David Newman tarafından tekrar düzenlenmiştir. Arşiv, dünya genelinde, bilgisayar öğrenmesi alanında çalışan araştırmacıların kullandığı en temel kaynak durumundadır.

5.5.1. Monk problemleri ile deneyler ve performans karşılaştırmaları

Monk problemleri, bilgisayar öğrenme algoritmaları alanında çalışan araştırmacıların geliştirdikleri algoritmaları, uluslararası düzeyde karşılaştırılmalarına ilk kez zemin hazırlamıştır. Problemler, Altı karakteristiği barındıran eğitim ve test setlerinden oluşan bir koleksiyondur. Setlerin yapısı,

<sınıf>: <değer1> <değer2> <değer3> <değer4> <değer5> <değer6>

olarak tasarlanmıştır. Burada <değerN>, karakteristik#N'ye ait değeri göstermektedir. <sınıf> ise, veri setindeki örneklerin ait oldukları sınıflara bağlı olarak 0 veya 1 değerini almaktadır. Veri setlerini oluşturan karakteristiklerin değerleri ise aşağıda gösterilen değerleri alırlar.

karakteristik#1: {1,2,3}

karakteristik#2: {1,2,3}

karakteristik#3: {1,2}

karakteristik#4: {1,2,3}

karakteristik#5: {1,2,3,4}

karakteristik#6: {1,2}

Bu çerçevede, monk1 problemine ait eğitim setindeki 10 örnek aşağıda gösterilmiştir.

1 1 1 2 1 2 2

1 1 1 2 2 3 1

1 1 1 2 2 4 1

1 1 1 2 3 1 2

1 1 2 1 1 1 2

2 1 2 1 1 2 1

2 1 2 1 1 3 1

2 1 2 1 1 4 2

1 1 2 1 2 1 1

2 1 2 1 2 3 1

IREM algoritması, Monk1, Monk2 ve Monk3 örnek setleri kullanılarak, günümüzdeki mevcut iyi bilinen bilgisayar öğrenmesi algoritmaları ile karşılaştırılmıştır. Karşılaştırma işlemi, "Experiment Databases for Machine Learning"[107] ismindeki internet tabanlı program kullanılarak gerçekleştirilmiştir. 2009 yılında düzenlenen "The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)"[108] konferansında ödül kazanan uygulama, bilgisayar öğrenmesi algoritma geliştiricileri için güvenilir ve işlevsel bir ortam sağlamaktadır. Algoritmaların ürettikleri doğruluk oranları, aşağıdaki tablolarda bulunmaktadır.

Monk 1 problemi

Monk 1 test seti ile gerçekleştirilen deney sonuclarında, doğruluk oranları kriterine göre değerlendirildiğinde IREM en iyi sonucu (%100) üreten algoritmaların içinde yer aldığı görülmektedir. Bu problemde en düşük doğruluk oranının %53 olduğu belirlenmiştir.

Tablo 5.8. Monk 1 problemi, algoritmaların doğruluk oranları

GOSE	1,00		VFI	0,97
ADTree	1,00		IBk	0,97
DecisionTable	1,00		AODE	0,96
ClassificationViaRegression	1,00		NaiveBayes	0,96
PART	1,00		NaiveBayesUpdateable	0,96
MultilayerPerceptron	1,00		NaiveBayesSimple	0,96
LMT	1,00		LWL	0,96
SimpleLogistic	1,00		LogitBoost	0,95
MultiClassClassifier	1,00		AdaBoostM1	0,95
NNge	1,00		Bagging	0,94
REPTree	1,00		MultiBoostAB	0,94
Id3	1,00		Ridor	0,93
Logistic	1,00		Winnow	0,92
Prism	1,00		SMO	0,87
Decorate	1,00		NIC	0,86
RBDT	1,00		RandomTree	0,85
IREM	**1,00**		OneR	0,81
RandomForest	0,99		DecisionStump	0,80
NBTree	0,99		IB1	0,79
J48	0,99		ConjunctiveRule	0,78
RandomCommittee	0,98		ZeroR	0,56
JRip	0,98		Vote	0,56
OrdinalClassClassifier	0,98		HyperPipes	0,56
FilteredClassifier	0,98		RacedIncrementalLogitBoost	0,54
LBR	0,98		CVParameterSelection	0,54
AttributeSelectedClassifier	0,98		MultiScheme	0,54
ThresholdSelector	0,98		RBFNetwork	0,54
REX	0,97		StackingC	0,53
BayesNet	0,97		Stacking	0,53
VotedPerceptron	0,97			

Monk 2 problemi

Monk 2 eğitim seti üzerindeki test sonuçları incelendiğinde, tüm algoritmaların doğruluk oranlarında düşme olduğu tespit edilmiştir. Bunun en önemli sebebi, Monk 2 eğitim setinin yapısındaki gürültülerdir. IREM algoritması tüm algoritmalar arasında %77 ile en iyi dördüncü doğruluk oranına sahiptir.

Tablo 5.9. Monk 2 problemi, algoritmaların doğruluk oranları

MultilayerPerceptron	0,90		REPTree	0,65
GOSE	0,84		MultiBoostAB	0,65
ClassificationViaRegression	0,82		HyperPipes	0,65
IREM	**0,77**		ZeroR	0,65
REX	0,76		Vote	0,65
LMT	0,71		JRip	0,65
PART	0,70		AttributeSelectedClassifier	0,64
ADTree	0,69		J48	0,64
NNge	0,69		SimpleLogistic	0,64
SMO	0,69		MultiClassClassifier	0,64
NIC	0,69		Ridor	0,63
LogitBoost	0,68		DecisionStump	0,63
RBDT	0,67		NBTree	0,63
StackingC	0,67		VotedPerceptron	0,62
Stacking	0,67		LBR	0,61
BayesNet	0,66		AODE	0,61
RBFNetwork	0,66		LWL	0,60
OneR	0,66		Logistic	0,60
RacedIncrementalLogitBoost	0,66		NaiveBayes	0,59
MultiScheme	0,66		NaiveBayesUpdateable	0,59
ConjunctiveRule	0,66		NaiveBayesSimple	0,59
CVParameterSelection	0,66		Decorate	0,58
ThresholdSelector	0,66		IBk	0,57
Bagging	0,65		Winnow	0,56
DecisionTable	0,65		IB1	0,54
OrdinalClassClassifier	0,65		RandomTree	0,53
FilteredClassifier	0,65		Prism	0,49
AdaBoostM1	0,65			

Monk 3 problemi

Monk 3 probleminde IREM algoritması %93 doğruluk oranı ile üst kategorideki algoritmalar arasında yer almıştır. Monk problemlerinde elde edilen sonuçlar, “hiçbir algoritmanın tüm veri setlerinde en üst performansı gösteremeyeceği” şeklinde özetlenebilecek “no free lunch” teoremini destekler mahiyette gerçekleşmiştir. Örneğin, prism ve decorate gibi algoritmalar monk 1 ve monk 3’te en yüksek seviyede doğruluk derecelerine sahipken, monk 2 deney setinde %49 ve %58 ile en düşük doğruluk oranlarında kalmışlardır.

Tablo 5.9 Monk 3 problemi, algoritmaların doğruluk oranları

NBTree	1,00		NaiveBayesUpdateable	0,83
PART	1,00		NaiveBayesSimple	0,83
NNge	1,00		Ridor	0,83
DecisionTable	1,00		LWL	0,83
LBR	1,00		HyperPipes	0,83
Decorate	1,00		MultiBoostAB	0,83
MultilayerPerceptron	1,00		AttributeSelectedClassifier	0,83
IBk	1,00		AdaBoostM1	0,82
OrdinalClassClassifier	1,00		LogitBoost	0,82
FilteredClassifier	1,00		VFI	0,82
Id3	1,00		Logistic	0,81
LMT	1,00		DecisionStump	0,81
Prism	1,00		SMO	0,79
RBDT	1,00		IB1	0,78
ClassificationViaRegression	0,97		ThresholdSelector	0,78
RandomForest	0,97		MultiClassClassifier	0,77
RandomCommittee	0,95		ConjunctiveRule	0,77
NIC	0,95		OneR	0,76
GOSE	0,94		BayesNet	0,75
REX	0,94		VotedPerceptron	0,73
IREM	**0,93**		Winnow	0,71
REPTree	0,93		ZeroR	0,58
AODE	0,93		Vote	0,58
JRip	0,92		RacedIncrementalLogitBoost	0,52
J48	0,87		CVParameterSelection	0,52
RandomTree	0,87		MultiScheme	0,52
ADTree	0,87		RBFNetwork	0,49
SimpleLogistic	0,85		StackingC	0,49
Bagging	0,84		Stacking	0,49
NaiveBayes	0,83			

5.6. Sonuçlar

Algoritmanın tanmında belirtildiği gibi, kural tabanı oluşturulurken, kuralları meydana getiren karakteristik-değer ikililerinden bilgi değeri yüksek olanlara öncelik verilmesi hedeflenmiştir. Bu bağlamda karakteristik-değer ikililerinin maliyetleri hesaplanmıştır. Maliyeti düşük olanlar bilgi değeri yüksek olduğu kabul edilmiştir. Maliyet fonksiyonlarının hesaplanmasında karakteristik-değer ikililerinin veri seti üzerindeki koşullu olasılıkları ve entropileri kullanılarak yeni bir yaklaşım

getirilmiştir. Yukarıdaki deney sonuçları incelendiğinde, IREM algoritmasının mevcut algoritmalarla rekabet edebilecek bir doğruluk yüzdesine sahip olduğu görülmektedir.

BÖLÜM 6. keREM (ke RULE EXTRACTION METHOD) ALGORİTMASI

6.1. Giriş

Bu bölümde yeni geliştirilen bir bilgisayar öğrenme algoritması olan keREM sunulacaktır. Bir önceki bölümde olduğu gibi, algoritmanın özellikleri ayrıntılı olarak verilerek ve örnek bir veri seti üzerinden kural üretme prosedürü açıklanacaktır. Ayrıca, gerçek veri setleri yardımı ile algoritmanın performansı gösterilecektir.

6.2. Algoritmanın Tanımı

keREM algoritması, IREM ile bir çok açıdan ortak özellikler taşımaktadır. keREM algoritması da, bilgisayar öğrenmesi alanında, danışmanlı öğrenme algoritmaları kategorisinde, kapsama yöntemini kullanarak EĞER-İSE (IF-THEN) yapısında genel kurallar üreten yöntemler arasında yer almaktadır. Veri setinden, bilgi değeri yüksek veriler seçilerek genel kurallar çıkarılması hedefi keREM algoritması içen de geçerli bir yaklaşımdır. keREM ve IREM algoritmalarının ayrıldığı nokta, verilerin bilgi değerlerini belirlemek için IREM maliyet fonksiyonu, keREM ise kazanç fonksiyonu kullanmasıdır. IREM'de hesaplanan maliyet fonksiyonu, karakteristik-değer ikililerinin olasılık dağılımları ve entropileri kullanır, keREM'nin kullandığı kazanç fonksiyonu ise, bir sonraki kısımda ayrıntıları ile incelenecek olan, koşullu olasılık değerleri, sınıflara dağılım oranı ve karakteristik sınıflandırma gücü parametreleri yardımı ile hesaplanmaktadır. keREM'de hesaplanan bu kazanç fonksiyonuna özel olarak kurallaşma eğilimi (**ke,** aynı zamanda algoritmanın ilk iki karakterini oluşturmaktadır.) ismi verilmiştir.

Gerçek veri setleri ile yapılan deneyler göstermiştir ki, farklı parametreler kullanılarak hesaplanan karakteristik-değer ikililerine ait maliyet değerlerinin

sıralamaları aynı olmaktadır. Bundan dolayı, kurallarında bilgi değeri yüksek karakteristik-değer ikililerine öncelik tanıma yaklaşınmını kulanan IREM ve keREM algoritmaları aynı kural tabanını oluşturmaktadır. Dolayısı ile algoritmaların doğruluk oranları aynıdır.

keREM algoritmasının işleyişine geçmeden önce, algoritmanın temelini oluşturan kazanç fonksiyonu üzerinde durulacaktır.

6.3. Kazanç Fonksiyonu (Kurallaşma Eğilimi-ke)

IREM algoritmasında belirtildiği gibi, kural oluşturma prosedürü içerisinde, bilgi değeri yüksek karakter-değer ikililerinin seçimine öncelik verilmektedir. Bilgi değerinin hesaplanabilmesi için kazanç fonksiyonlarında üretilen sonuçlar baz alınmaktadır. Kazancı yüksek karakter-değer ikililerinin bilgi değerlerinin yüksek olduğu prensibi üzerinden hareket edilmektedir. Bu bağlamda, kazanç fonksiyonunun hesaplanmasını sağlayan parametreler aşağıda sunulmaktadır.

Sınıf bazındaki koşullu olasılıklar;

$P(V_C)$, veri setindeki her değerin sınıf bazındaki koşullu olasılıkları,

$|V_C|$, C sınıfında olan V değerlerinin sayısı,

$|V|$, Veri setindeki tüm V değerlerinin sayısı olsun.

Buna göre sınıf bazındaki koşullu olasılıklar,

$$P(V_C) = |V_C| / |V| \quad (6.1)$$

şeklinde hesaplanır.

Sınıflara dağılım oranı;

DR_V, bir V değerinin sınıflara dağılım oranı,

$C=\{ c_1, c_2, c_3,\ldots, c_m \}m$, sınıf sayısı,

D_V, bir V değerinin m sınıftan kaç tanesinde geçtiğini göstersin.

Buna göre herbir değerin sınıflara dağılım oranı,

$$DR_V = (1 / D_V) \quad (6.2)$$

şeklinde hesaplanır.

Karakteristik sınıflandırma gücü;

POW_A, bir karakteristiğin sınıflandırma gücü,

$C=\{ c_1, c_2, c_3,\ldots, c_m \}m$, sınıf sayısı,

$A=\{a_1, a_2, a_3,\ldots, a_y\}$ y, veri setindeki karakteristik sayısı,

$A_V=\{Av_1, Av_2, Av_3,\ldots, Av_n\}$ n, bir karakteristiğe ait tüm değerlerin sayısı,

$P(AV_n)$ max, bir karakteristiğe ait herhangi bir değerin tüm sınıflardaki olasılıklarının en büyük değeri olsun,

Buna göre bir karakteristiğin sınıflandırma gücü,

$$POW_A = (P(AV_1)_{max} + P(AV_2)_{max} + \cdots + P(AV_n)_{max})/m \quad (6.3)$$

Şeklinde hesaplanır. Yukarıdaki parametreleri kullanılarak gerçekleştirilen, belirli bir sınıfataki değerin kurallaşma eğilimini ($ke_{V-C)}$ gösteren kazanç fonksiyonu şu şekilde hesaplanır.

$P(V_C)$, veri setindeki her değerin sınıf bazındaki koşullu olasılıkları,

DR_V, bir V değerinin sınıflara dağılım oranı,

POW_A, bir karakteristiğin sınıflandırma gücü,

$AVG(P(V_C), DR_V, POW_A)$, parametrelerin ortalaması olsun.

$$ke_{V-C} = \begin{cases} 0, \ P(V_c) = 0 \\ 1, \ P(V_c) = 1 \\ AVG(\mathrm{P}(\mathrm{V_C}), \mathrm{DR_V}, POW_A), \ 0 < P(V_c) < 1 \end{cases} \quad (6.4)$$

6.5. keREM Algoritmasının İşleyişi

Geliştirilen keREM algoritmasının işleyişi aşağıda adım adım açıklanmıştır. Algoritmada geçen koşullu olasılık değerleri, sınıflara dağılım oranı, karakteristik sınıflandırma gücü ve maliyet fonksiyonu gibi kavramlar örnek uygulama vasıtasıyla ayrıntılı olarak anlatılacaktır.

(N: karakteristik sayısı, n : kombinasyon sayısı olmak üzere)

Adım-1. Verilen örnek setindeki tüm karakteristik değerlerinin, sınıf bazında olasılık dağılımları (koşullu olasılık değerleri) ve sınıflara dağılım oranları hesaplanır.

Adım-2. Veri setindeki her karakteristiğin sınıflandırma gücü hesaplanır.

Adım-3. Koşullu olasılık değerleri, sınıflara dağılım oranı, karakteristik sınıflandırma gücü parametreleri kullanılarak, tüm karakteristik-değer ikililerinin kazançları hesaplanır.

Adım-4. Her örnekteki değerlerin tekli (n=1) kombinasyonları ele alınır. n=1 için entropisi sıfır olan değerler tek başlarına kural oluşturabilirler. Bu değerler kural haline getirilir (Bu değerlerden aynı örneklere karşılık gelen kurallardan, en çok örneği kapsayan kural geçerli olarak kabul edilir). Sınıflandırılan örnekler işaretlenir.

Adım-5. Adım-8'e git.

Adım-6. Sınıflandırılmamış örnekler sırasıyla ele alınır. Aktif örneğin sınıfına bağlı olarak, sınıf bazındaki kazancı en yüksek karakteristik değerden başlamak şartı ile, karakteristik-değer ikililerinden n'li kombinasyonlar oluşturulur.

Adım-7. Her kombinasyon örnek setindeki tüm örneklere uygulanır. n adet kombinasyondan oluşan değerlerden tek bir sınıfa karşılık gelenleri kural haline getirilir. Sınıflandırılan örnekler işaretlenir.

Adım-8. Eğer tüm örnekler sınıflandırılmış ise Adım-11'ye gidilir.

Adım-9. n=n+1 işlemi yapılır.

Adım-10. Eğer n<N ise Adım-6'ye gidilir.

Adım-11. Oluşturulan kural tabanı, aynı örnekleri temsil eden kurallar arasından en genel kurallar seçilirek optimize edilir.

Adım-12. SON

6.5. keREM ile Örnek Uygulama

Dikkat edileceği üzere, keREM algoritmasının tanımı kısmında, IREM ve keREM algoritmalarının aynı kural tabanını ürettiklerine ve doğruluk oranlarının eşit olduğuna işaret edilmişti. keREM ile uygulama yaparken, bu özelliği de gösterme adına tekrar mevsimler eğitim seti üzerinden açıklamalar gerçekleştirilecektir.

Tablo 6.1. Mevsimler eğitim seti

Örnek No	Hava	Ağaçlar	Sıcaklık	Mevsim
1	Yağmurlu	Yapraksız	Normal	Sonbahar
2	Yağmurlu	Yapraksız	Düşük	Kış
3	Karlı	Yapraksız	Düşük	Kış
4	Güneşli	Yapraksız	Düşük	Kış
5	Yağmurlu	Sarı	Normal	Sonbahar
6	Yağmurlu	Yeşil	Yüksek	Yaz
7	Yağmurlu	Yeşil	Normal	İlkbahar
8	Güneşli	Yeşil	Normal	İlkbahar
9	Güneşli	Yeşil	Yüksek	Yaz
10	Güneşli	Sarı	Normal	Sonbahar
11	Karlı	Yeşil	Düşük	Kış

Tablo 6.1'de verilen örnek seti 11 örnekten (m = 11), üç karakteristik (Hava, Ağaçlar, Sıcaklık) ve dört sınıftan (İlkbahar, Yaz, Sonbahar, Kış) oluşmaktadır. Örnek setinde yer alan karakteristikler ve bunlara ait değerler aşağıda verilmiştir:

Karakteristik	**Değerler**
Hava	Yağmurlu, Karlı, Güneşli
Ağaçlar	Yapraksız, Sarı, Yeşil
Sıcaklık	Normal, Düşük, Yüksek

Adım-1. Verilen örnek setindeki tüm karakteristik değerlerinin, sınıf bazında olasılık dağılımları hesaplanır.

$P(\text{yağmurlu}_{\text{sonbahar}}) = |\text{yağmurlu}_{\text{sonbahar}}| / |\text{yağmurlu}|$

$P(\text{yağmurlu}_{\text{sonbahar}}) = 2/5 = 0{,}40$

Diğer değerlere ait olasılık dağılımlar da benzer şekilde hesaplanır. Tablo 6.1'de tüm değerlerin sınıf bazındaki olasılık dağılımları gösterilmiştir.

Tablo 6.2. Değerlerin sınıf bazındaki koşullu olasılıkları

Karakteristik	**Değer**	**Sonbahar**	**Kış**	**Yaz**	**İlkbahar**
Hava	Yağmurlu	0,40	0,20	0,20	0,20
	Karlı	0,00	1,00	0,00	0,00
	Güneşli	0,25	0,25	0,25	0,25
Ağaçlar	Yapraksız	0,25	0,75	0,00	0,00
	Sarı	1,00	0,00	0,00	0,00
	Yeşil	0,00	0,20	0,40	0,40
Sıcaklık	Normal	0,60	0,00	0,00	0,40
	Düşük	0,00	1,00	0,00	0,00
	Yüksek	0,00	0,00	1,00	0,00

Bu adımda ayrıca, veri setindeki her değerin sınıflara dağılım oranı hesaplanır. “Yağmurlu” değeri ele alınacak olursa, yağmurlu değeri 4 sınıftada geçmektedir. Bundan dolayı, yağmurlu değerinin sınıflara dağılım oranı DRyağmurlu,

$DR_{yağmurlu} = 1 / 4 = 0,25$

“Karlı” değeri ele alındığında; karlı değeri sadece kış mevsiminde geçmektedir. Bu suretle DRkarlı,

$DR_{karlı} = 1 / 1 = 1$

Benzer şekilde tüm değerlerin sınıflara dağılım oranları hesaplandığında sonuçlar Tablo 6.3’teki gibi olacaktır.

Tablo 6.3. Değerlerin sınıflara dağılım oranları

Karakteristik	**Değer**	**Sınıflara Dağılım Oranı**
Hava	Yağmurlu	0,25
	Karlı	1
	Güneşli	0,25
Ağaçlar	Yapraksız	0,50
	Sarı	1
	Yeşil	0,33
Sıcaklık	Normal	0,50
	Düşük	1
	Yüksek	1

Adım-2. Veri setindeki her karakteristiğin sınıflandırma gücü hesaplanır.

“Hava ” karakteristiği ele alındığında, hava karakteristiğinin yağmurlu değeri için sınıflar bazındaki en büyük değeri;

Sonbahar sınıfında: **0,40**

Kış sınıfında: 0,20

Yaz sınıfında: 0,20

İlkbahar sınıfnda:0,20

Yukarıdaki değerlere göre,

$P(HAVA_{yağmurlu})_{max}$ =0, 40 tır.

Benzer şekilde,

$P(HAVA_{karlı})_{max}$ = 1

$P(HAVA_{güneşli})_{max}$ = 0,25

olarak belirlenir. Yukarıdaki değerlere göre hava karakteristiğinin sınıflandırma gücü,

$$POW_{hava} = \left(P(HAVA_{yağmurlu})_{max} + P(HAVA_{karlı})_{max} + P(HAVA_{güneşli})_{max}\right)/3$$

$$POW_{hava} = (0{,}40 + 1 + 0{,}25)/3$$

$$POW_{hava} = 0{,}55$$

Ağaçlar ve sıcaklık karakteristikleri için yukarıdaki şekilde hesaplama yapıldığında Tablo 6.4 oluşturulur.

Tablo 6.4. Karakteristiğin sınıflandırma gücü

Karakteristik	**Değer**	**Karakteristiğin Sınıflandırma Gücü**
Hava	Yağmurlu	0,55
	Karlı	0,55
	Güneşli	0,55
Ağaçlar	Yapraksız	0,71
	Sarı	0,71
	Yeşil	0,71
Sıcaklık	Normal	0,86
	Düşük	0,86
	Yüksek	0,86

Adım-3. Koşullu olasılık değerleri, sınıflara dağılım oranı, karakteristik sınıflandırma gücü parametreleri kullanılarak, tüm karakteristik-değer ikililerinin kazançları hesaplanır.

Örnek olarak “hava” karakteristiğine ait “yağmurlu” değerinin “sonbahar” sınıfında ele alınarak kurallaşma eğilimi hesaplanırsa,

$$ke_{V-C} = \begin{cases} 0, & P(V_c) = 0 \\ 1, & P(V_c) = 1 \\ AVG(\mathrm{P}(V_c), \mathrm{DR}_V, POW_A), & 0 < P(V_c) < 1 \end{cases}$$

$P(\text{Yağmurlu}_{\text{sonbahar}}) = 0{,}40$

$0 < P(\text{Yağmurlu}_{\text{sonbahar}}) < 1$ şartı doğru olduğundan,

$ke_{\text{yağmurlu-sonbahar}} = AVG(P(\text{Yağmurlu}_{\text{sonbahar}}), DR_{\text{yağmurlu}}, POW_{\text{hava}})$

$ke_{\text{yağmurlu- sonbahar}} = AVG(0.40, 0.25, 0.55)$

$ke_{\text{yağmurlu- sonbahar}} = 0{,}40$

olarak hesaplanır. Benzer şekilde diğer değerlere ait kurallaşma eğilimleri sınıflar bazında hesaplandığında aşağıdaki tablo oluşturulur.

Tablo 6.5. Sınıf bazlı kurallaşma eğilimleri

Özellik	Değer	Sonbahar	Kış	Yaz	İlkbahar
Hava	Yağmurlu	0,40	0,33	0,33	0,33
	Karlı	0,00	1,00	0,00	0,00
	Güneşli	0,35	0,35	0,35	0,35
Ağaçlar	Yapraksız	0,49	0,65	0,00	0,00
	Sarı	1,00	0,00	0,00	0,00
	Yeşil	0,00	0,41	0,48	0,48
Sıcaklık	Normal	0,65	0,00	0,00	0,59
	Düşük	0,00	1,00	0,00	0,00
	Yüksek	0,00	0,00	1,00	0,00

Aşağıdaki tabloda açıkça görülebileceği gibi, veri setindeki her değerin sınıf bazındaki bilgi değerleri birbirinden farklıdır. Algoritmanın temelini oluşturan bu anlayış çerçevesinde, Adım-4’ten itibaren kural üretme prosedüründe Tablo 6.6’daki

sıralamaya bağlı kalınarak, karakteristik-değer ikililerinin seçimi gerçekleştirilecektir.

Tablo 6.6. Değerlerin sınıf bazında ke_V'ye göre sıralaması

Sonbahar		**Kış**		**Yaz**		**İlkbahar**	
Değer	**ke_V**	**Değer**	**ke_V**	**Değer**	**ke_V**	**Değer**	**ke_V**
Sarı	1,00	Karlı	1,00	Yüksek	1,00	Normal	0,59
Normal	0,65	Düşük	1,00	Yeşil	0,48	Yeşil	0,48
Yapraksız	0,49	Yapraksız	0,65	Güneşli	0,35	Güneşli	0,35
Yağmurlu	0,40	Yeşil	0,41	Yağmurlu	0,33	Yağmurlu	0,33
Güneşli	0,35	Güneşli	0,35	Karlı	0,00	Karlı	0,00
Karlı	0,00	Yağmurlu	0,33	Yapraksız	0,00	Yapraksız	0,00
Yeşil	0,00	Sarı	0,00	Sarı	0,00	Sarı	0,00
Düşük	0,00	Normal	0,00	Normal	0,00	Düşük	0,00
Yüksek	0,00	Yüksek	0,00	Düşük	0,00	Yüksek	0,00

Adım-4: Tablo 6.6'daki örnek setindeki kurallaşma eğilimleri (ke) 1 (bir) olan değerler,

Sıcaklık karakteristiği için; Düşük ve Yüksek

Ağaçlar karakteristiği için; Sarı

Hava karakteristiği için; Karlı

Bu değerler tekli kombinasyonlara göre aşağıdaki kuralları oluştururlar:

Kural-1: EĞER Sıcaklık Düşük İSE Mevsim KIŞ

Bu kural ile 2, 3, 4 ve 11 nolu örnekler sınıflandırılır. Bu örneklerin tümü sadece KIŞ sınıfına aittir.

Kural-2: EĞER Sıcaklık Yüksek İSE Mevsim YAZ

Bu kural ile 6 ve 9 nolu örnekler sınıflandırılır. Bu örnekler sadece YAZ sınıfına aittir.

Kural-3: EĞER Ağaçlar Sarı İSE Mevsim SONBAHAR

Bu kural ile 5 ve 10 nolu örnekler sınıflandırılır. Bu örnekler sadece Sonbahar sınıfına aittir.

Kural-4: EĞER Hava Karlı İSE Mevsim KIŞ

Bu kural ile 3 ve 11 nolu örnekler sınıflandırılır. Ancak 3 ve 11 nolu örnekler daha önce Kural-1 ile sınıflandırıldığı için bu örnekler için tekrar kural oluşturulmaz. Bu kural geçersiz olarak kabul edilir.

Bu işlemlerden sonra Adım-8'e gidilir. Tüm örnekler sınıflandırılmadığından dolayı Adım-9'daki n=n+1 işlemi ile n'nin değeri bir artırılarak iki olur ve Adım-6'ya gidilir.

Adım-6: Sınıflandırılmamış örnekler sırasıyla ele alınır. Aktif örneğin sınıfına bağlı olarak, sınıf bazındaki kurallaşma eğilimleri en yüksek karakteristik değerinden başlamak şartı ile, karakteristik değerlerinden n'li kombinasyonlar oluşturulur.

Sınıflandırılamayan örnekler olarak 1, 7 ve 8 nolu örnekler bulunmaktadır. Bu örnekler Tablo 6.7'de gösterilmiştir.

Tablo 6.7. Birinci aşama sonunda sınıflandırılamayan örnekler

Örnek No	**Hava**	**Ağaçlar**	**Sıcaklık**	**Mevsim**
1	Yağmurlu	Yapraksız	Normal	Sonbahar
7	Yağmurlu	Yeşil	Normal	İlkbahar
8	Güneşli	Yeşil	Normal	İlkbahar

6. Adımdaki şarta bağlı olarak, birinci örnekteki değerlerin ikili kombinasyonları oluşturulur. 1. Örnek sonbahar sınıfına ait olduğundan, değerlerin sonbahar sınıfındaki kurallaşma eğilimleri dikkate alınır.

$ke_{\text{normal- sonbahar}} = 0{,}65$

$ke_{\text{yapraksız-sonbahar}} = 0{,}49$

$ke_{\text{yağmurlu-sonbahar}} = 0{,}40$

Kurallaşma eğilimi değerlerine bağlı olarak sırasıyla,
{Normal,Yapraksız}, {Normal, Yağmurlu}, {Yapraksız, Yağmurlu}

ikili kombinasyonlar oluşturulur.

Adım-7: İkili kombinasyonların ilki olan {Normal,Yapraksız} değerleri için tüm örnek setine bakılır. Bu değerler ile sadece 1 nolu örnek sınıflandırılabilmektedir.

Kural-4 : EĞER Sıcaklık Normal VE Ağaçlar Yapraksız İSE Mevsim SONBAHAR

Sınıflandırılamayan örnekler olarak 7 ve 8 nolu örnekler kalmaktadır. Yedinci örnekteki değerlerin ikili kombinasyonları oluşturulur.

$ke_{\text{normal-ilkbahar}} = 0{,}59$

$ke_{\text{yeşil- ilkbahar}} = 0{,}48$

$ke_{\text{yağmurlu-ilkbahar}} = 0{,}33$

Kurallaşma eğilimi değerlerine bağlı olarak sırasıyla,

{Normal,Yeşil}, {Normal,Yağmurlu} ve {Yeşil, Yağmurlu}

ikili kombinasyonlar oluşturulur.

{Normal, Yeşil} değerleri 7 ve 8 nolu örneklerde sadece bir sınıfta geçtiklerinden kural haline getirilir.

Kural-5 : EĞER Sıcaklık Normal VE Ağaçlar Yeşil İSE Mevsim İLKBAHAR

Sınıflandırılacak başka örnek kalmadığından dolayı işlemler tamamlanır. Çıkarılan kurallar Tablo 6.8'de gösterilmiştir.

Tablo 6.8. Mevsimler eğitim setii için üretilen kurallar

Kural No	Kural Tanımı
1	**EĞER** Sıcaklık=Düşük **İSE** Mevsim=KIŞ
2	**EĞER** Sıcaklık=Yüksek **İSE** Mevsim=YAZ
3	**EĞER** Ağaçlar=Sarı **İSE** Mevsim=SONBAHAR
4	EĞER Sıcaklık=Normal VE Ağaçlar=Yeşil **İSE** Mevsim=İLKBAHAR
5	EĞER Sıcaklık=Normal VE Ağaçlar=Yapraksız **İSE** Mevsim=SONBAHAR

6.6. Deneyler ve Performans Karşılaştırmaları

Beşinci bölümde incelendiği gibi, IREM algoritması verilerin bilgi değerlerini ölçerken kullandığı maliyet fonksiyonu, olasılık ve entropi tabanlıdır. keREM algoritmasının kazanç fonksiyonun hesaplama şekli çok farklı olmasına rağmen, veri setindeki karakteristik-değer ikililerinin sınıf bazındaki bilgi değerleri aynı sıralamaya sahip oldukları görülmektedir (Tablo 6.9, Tablo 6.10, Tablo 6.11, Tablo 6.12).

Tablo 6.9. IREM ve keREM'in "sonbahar" sınıfında ürettiği maliyet ve kazanç

Karakteristik	Değer	Maliyet (IREM)	Kazanç (keREM)
Ağaçlar	Sarı	0,000	1,00
Sıcaklık	Normal	0,565	0,65
Ağaçlar	Yapraksız	0,882	0,49
Hava	Yağmurlu	2,114	0,40
Hava	Güneşli	2,701	0,35
Hava	Karlı	-1	0,00
Ağaçlar	Yeşil	-1	0,00
Sıcaklık	Düşük	-1	0,00
Sıcaklık	Yüksek	-1	0,00

Tablo 6.10. IREM ve keREM'in "kış" sınıfında ürettiği maliyet ve kazanç

Karakteristik	Değer	Maliyet (IREM)	Kazanç (keREM)
Hava	Karlı	0,000	1,00
Sıcaklık	Düşük	0,000	1,00
Ağaçlar	Yapraksız	0,294	0,65
Ağaçlar	Yeşil	2,007	0,41
Hava	Güneşli	2,701	0,35
Hava	Yağmurlu	2,818	0,33
Ağaçlar	Sarı	-1	0,00
Sıcaklık	Normal	-1	0,00
Sıcaklık	Yüksek	-1	0,00

Tablo 6.11. IREM ve keREM'in "yaz" sınıfında ürettiği maliyet ve kazanç

Karakteristik	Değer	Maliyet (IREM)	Kazanç (keREM)
Sıcaklık	Yüksek	0,000	1,00
Ağaçlar	Yeşil	1,505	0,48
Hava	Güneşli	2,701	0,35
Hava	Yağmurlu	2,818	0,33
Hava	Karlı	-1	0,00
Ağaçlar	Yapraksız	-1	0,00
Ağaçlar	Sarı	-1	0,00
Sıcaklık	Normal	-1	0,00
Sıcaklık	Düşük	-1	0,00

Tablo 6.12. IREM ve keREM'in "ilkbahar" sınıfında ürettiği maliyet ve kazanç

Karakteristik	Değer	Maliyet (IREM)	Kazanç (keREM)
Sıcaklık	Normal	0,847	0,59
Ağaçlar	Yeşil	1,505	0,48
Hava	Güneşli	2,701	0,35
Hava	Yağmurlu	2,818	0,33
Hava	Karlı	-1	0,00
Ağaçlar	Yapraksız	-1	0,00
Ağaçlar	Sarı	-1	0,00
Sıcaklık	Düşük	-1	0,00
Sıcaklık	Yüksek	-1	0,00

Mevsimler eğitim seti üzerinde yapılan uygulamalarda da açığa çıktığı gibi, IREM ve keREM algoritmalarının ürettiği kural tabanları aynı olmaktadır. Bu sonuç, algoritmaların doğruluk oranlarının aynı olacağına işaret etmektedir. Gerçek veri setleri ile yapılan uygulamalardaki doğruluk oranlarıda aynı olduğu görülmüştür.

Tablo 6.13. Algoritmaların monk problemlerindeki doğruluk oranları ve ortalamaları

Algoritma	Monk1 D.O(%)	Monk2 D.O(%)	Monk3 D.O(%)	Ortalama D.O(%)
MultilayerPerceptron	1,00	0,90	1,00	0,97
ClassificationViaRegression	1,00	0,82	0,97	0,93
GOSE	1,00	0,85	0,94	0,93
keREM - IREM	**1,00**	**0,77**	**0,93**	**0,90**
LMT	1,00	0,71	1,00	0,90
PART	1,00	0,70	1,00	0,90
REPTree	1,00	0,76	0,93	0,90
NNge	1,00	0,69	1,00	0,90
RBDT	1,00	0,67	1,00	0,89
DecisionTable	1,00	0,65	1,00	0,88
FilteredClassifier	0,98	0,65	1,00	0,88
OrdinalClassClassifier	0,98	0,65	1,00	0,88
NBTree	0,99	0,63	1,00	0,87
LBR	0,98	0,61	1,00	0,86
Decorate	1,00	0,58	1,00	0,86
ADTree	1,00	0,69	0,87	0,85
JRip	0,98	0,65	0,92	0,85
SimpleLogistic	1,00	0,69	0,85	0,85
REX	0,97	0,63	0,94	0,85
IBk	0,97	0,57	1,00	0,84
NIC	0,86	0,69	0,95	0,83
J48	0,99	0,64	0,87	0,83
AODE	0,96	0,61	0,93	0,83
RandomForest	0,99	0,53	0,97	0,83
Prism	1,00	0,49	1,00	0,83
LogitBoost	0,95	0,68	0,82	0,82
AttributeSelectedClassifier	0,98	0,64	0,83	0,82
Bagging	0,94	0,65	0,84	0,81
AdaBoostM1	0,95	0,65	0,82	0,81
MultiBoostAB	0,94	0,65	0,83	0,81
MultiClassClassifier	1,00	0,64	0,77	0,80
ThresholdSelector	0,98	0,65	0,78	0,80
Ridor	0,93	0,64	0,83	0,80
Logistic	1,00	0,60	0,81	0,80
LWL	0,96	0,60	0,83	0,80
BayesNet	0,97	0,66	0,75	0,80
NaiveBayes	0,96	0,59	0,83	0,80
NaiveBayesSimple	0,96	0,59	0,83	0,80
NaiveBayesUpdateable	0,96	0,59	0,83	0,80
VotedPerceptron	0,97	0,65	0,73	0,78

Tablo 6.13.'ün devamı

SMO	0,87	0,67	0,79	0,78
DecisionStump	0,80	0,63	0,81	0,74
OneR	0,81	0,66	0,76	0,74
ConjunctiveRule	0,78	0,66	0,77	0,74
IB1	0,79	0,54	0,78	0,70
HyperPipes	0,56	0,65	0,83	0,68
MultiScheme	0,54	0,66	0,52	0,57
RacedIncrementalLogitBoost	0,54	0,66	0,52	0,57
CVParameterSelection	0,54	0,66	0,52	0,57
Vote	0,56	0,56	0,58	0,57
Stacking	0,53	0,67	0,49	0,56
StackingC	0,53	0,66	0,49	0,56
RBFNetwork	0,54	0,65	0,49	0,56

6.7. Sonuçlar

keREM algoritmasında da, kural tabanı oluşturulurken, kuralları meydana getiren karakteristik-değer ikililerinden bilgi değeri yüksek olanlara öncelik verilmesi hedeflenmiştir. Bu bağlamda karakteristik-değer ikililerinin değerleri, kurallaşma eğilimi ismi verilen kazanç fonksiyonu ile hesaplanmıştır. Kazancı yüksek olanlar bilgi değeri yüksek olduğu kabul edilmiştir. Kazanç fonksiyonlarının hesaplanmasında karakteristik-değer ikililerinin veri seti üzerindeki koşullu olasılıkları, sınıflara dağılım oranları ve karaktesirtiğin sınıflama gücü parametreleri kullanılarak yeni bir yaklaşım getirilmiştir. Yukarıdaki deney sonuçları incelendiğinde, keREM ve IREM algoritmalarının kullandıkları kazanç ve maliyet foksiyonlarının, karakteristik-değer ikililerinin sınıf bazında ürettiği değerlerin sıralamaları aynı olduğu tespit edilmiştir. keREM ve IREM algoritmalarının kural üretme yapılarından dolayı, ortaya çıkan kural tabanı, dolayısıyla test sonuçlarındaki doğruluk oranları aynı olduğu görülmüştür. Bununla beraber, keREM ve IREM'in, mevcut algoritmalarla rekabet edebilecek bir doğruluk yüzdesine sahip olduğuda saptanmıştır.

BÖLÜM 7. SONUÇLAR VE ÖNERİLER

Bu çalışmada, bilgisayar öğrenmesinde tümevarımsal olarak kural çıkarma işlemi gerçekleştirebilecek algoritmalar geliştirilmesi hedeflenmiştir. Kural üretme prosedüründe veri setinden doğrudan kural üretme yöntemini benimseyen, IREM ve keREM isimleri verilen algoritmalar, bilgisayar öğrenmesi altında mantıksal/sembolik algoritmalar kategorisinde yer almaktadır.

Geliştirilen her iki algoritmada kullanılan temel yaklaşım; veri seti içerisinde bulunan karakteristik-değer ikililerinden, bilgi değeri yüksek verilerin kurallarda yer almasını sağlamaktır. Bu açıdan, verilerin bilgi değerlerinin ölçülmesi üzerine odaklanılmıştır. IREM algoritmasında, verilerin olasılık dağılımları ve entropilerinden istifade edilerek maliyet fonksiyonu geliştirilmiştir.

Maliyet fonksiyonu;

$$M(V|C) = \begin{cases} 0, & P(V_c) = 1 \\ -1, & P(V_c) = 0 \\ k_{v,c} * E(V) + k_{v,c} * E(X), & 0 < P(V_c) < 1 \end{cases}$$

Maliyet fonksiyonuna göre, maliyeti düşük olan karakteristik-değer ikililerine öncelik verilerek kurallar üretilmiştir.

keREM algoritmasında ise, karakteristik-değer ikililerinin olasılık dağılımları, sınıf dağılım oranları ve karakteristik sınıflandırma gücü parametreleri kullanılarak, kurallaşma eğilimi (ke) ismi verilen kazanç fonksiyonu oluşturulmuştur.

Kazanç fonksiyonu;

$$ke_{V-C} = \begin{cases} 0, & P(V_c) = 0 \\ 1, & P(V_c) = 1 \\ AVG(\mathrm{P}(\mathrm{V_C}), \mathrm{DR_V}, POW_A), & 0 < P(V_c) < 1 \end{cases}$$

Kullanılan kazanç fonksiyonuna göre, kazancı yüksek verilere, üretilecek kurallarda öncelik tanınmıştır.

Geliştirilen IREM ve keREM fonksiyonlarının performanslarını ölçmek için, UCI bilgisayar öğrenmesi veri havuzunda bulunan deney setleri kullanılarak mevcut algoritmalar ile karşılaştırmaları yapılmıştır. Çıkan sonuçlar incelendiğinde IREM ve keREM algoritmalarını mevcut algoritmalarla rekabet edebilecek seviyede doğruluk oranına sahip oldukları görülmüştür.

Gelecekte yapılabilecek çalışmalar için aşağıdaki öneriler sunulmuştur;

IREM ve keREM algoritmalarının doğruluk oranlarını geliştirmek için yeni çalışmalar yapılabilir. Kullanılan maliyet ve kazanç fonksiyonlarında farklı düzenlemeler gerçekleştirilebilir.

Bilgisayar öğrenmesinde, mantıksal/sembolik kategorisindeki algoritmalar, sırasal kapsama (sequential covering) yaklaşımını kullanılarak kural üretmektedirler. Bu metoda göre, kural üretme prosedüründe, kurallarda yer almaya aday olabilecek karakteristik-değer ikililerini barındıran örneklerin kaybolduğu düşünülmektedir. Bundan dolayı, bu kaybı önlemek için doğrudan ulaşım (direct access) yaklaşımı ismi verilebilecek yönteme ait çalışmalar yapılabilir.

Geliştirilecek algoritmaların kural tabanındaki kural sayısının azaltılması, bununla beraber, kuralların kapsayıcılığının artırılmasına yönelik çalışmalar yapılabilir.
Bilgisayar öğrenme algoritmalarında, karakteristik seçme metodlarına alternatif olarak, keREM ve IREM'de kullanılan kazanç ve maliyet fonksiyonlarının kullanımı üzerinde çalışmalar gerçekleştirilebilinir.

Gerçek zamanlı uygulamalarda iyi performans gösterebilecek algoritmalar tasarlanabilir. Bunun için kural tabanını sürekli yenileyen ve optimize eden algoritmalar üzerinde çalışılabilir

KAYNAKLAR

[1] GUILFOYLE, C., Ten minutes to lay the foundations. Expert Systems User, 8, 16-19, 1986.

[2] LEECH, W.J., A rule-based process control method with feedback. Advances in Instrumentation, 41, 169-175, 1986.

[3] EVANS, B., FISHER, D., Overcoming Process Delays with Decision Tree Induction, IEEE Expert: Intelligent Systems and Their Applications, 9, 60-66, 1994.

[4] SUMATHI, S., SIVANANDAM, S.N., Introduction to Data Mining and its Applications, Springer-Verlag, Berlin, 2006.

[5] WITTEN, I., FRANK E., Data Mining - Practical Machine Learning Tools and Techniques, Morgan Kaufmann Publishers, San Francisco, 28, 2005.

[6] BAESENS, B., Developing intelligent systems for credit scoring using machine learning techniques. PhD thesis, K.U. Leuven, 2003.

[7] MARTENS, D., BAESENS, B., VAN GESTEL, T., VANTHIENEN, J., Comprehensible credit scoring models using rule extraction from support vector machines. European Journal of Operational Research, 183(3), 1466–1476, 2007.

[8] VAN GESTEL, T., BAESENS, B., VAN DIJCKE, P., A process model to develop an internal rating system: sovereign credit ratings. Decision Support Systems, 42(2), 1131–1151, 2006.

[9] KEIKHA, M., KHONSARI, A., OROUMCHIAN, F., Rich document representation and classification: An analysis, Knowledge-Based Systems 22, 67–71, 2009.

[10] DASGUPTA, A., Feature selection methods for text classification, Proceedings of the 13th ACMSIGKDD international conference on Knowledge discovery and data mining, 230 -239, 2007.

[11] AURANGZEB, K., BAHARUDIN, B., LEE, L., Review of Machine Learning Algorithms for Text-Documents Classification, Journal Of Advances In Informatıon Technology, 1, 2010.

[12] ÖZGÜR, S. O., SHAWE-TAYLOR, J., WEBER, G., Pattern analysis for the prediction of fungal pro-peptide cleavage sites. Computational Biology, 2007.

[13] HUYSMANS, J., BAESENS, B., MARTENS, D., DENYS, K., VANTHIENEN, J., New trends in data mining. Tijdschrift voor economie en Management, 50, 697–711, 2005.

[14] WANG, X., LI, A., JIANG, Z., AND FENG, H. Missing value estimation for DNA microarray gene expression data by Support Vector Regression imputation and orthogonal coding scheme. BMC Bioinformatics, 7, 32. 2006.

[15] LAMERS, S. et al.. Prediction of R5, X4, and R5X4 HIV-1 Coreceptor Usage with Evolved Neural Networks. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 5(2), 291–300. 2008.

[16] M. PAZZANI, S. MANI, AND W. SHANKLE. Acceptance by medical experts of rules generated by machine learning. Methods of Information in Medicine, 40(5):380–385, 2001.

[17] FISHER, S. et al. Genetic determinants of ulcerative colitis include the ECM1 locus and five loci implicated in Crohn's disease. Nature Genetics, 40, 710–712, 2008.

[18] LANGLAY, P., SIMON, H**.,** Applications of machine learning and rule induction, Communication of the acm, 38, 55-60, 1995.

[19] HUNT, E. B., MARIN, J., STONE, P. J., Experiments in induction, Academic Press, New York, 1966.

[20] QUINLAN, J. R., Learning efficient classification procedures and their application to chess end games, Machine Learning - An Artificial Intelligence Approach, Eds: Michalski; R.S., Carbonell, J.G. and Mitchell, T.M. (Eds), Tioga Publishing Co, Palo Alto. CA, 463-482, 1983.

[21] QUINLAN, J.R., Induction of decision trees, Machine Learning 1, Kluwer Academic Publishers, Boston, 81- 106, 1990.

[22] QUINLAN, J.R., C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA, 1993.

[23] BREIMAN, L.,et al. Classification and Regression Trees, Wadsworth International Group, Belmont, California, 1984.

[24] CRAWFORD, S. L., Extensions to the CART algorithm, Machine Learning and Uncertain Reasoning - Knowledge-Based Systems, 3, Gaines, B and Boose, J. (Eds), Academic Press, London, 15-35, 1990.

[25] ZHONG, N., DONG, J., OHSUGA, S., Rule discovery by soft induction techniques, Neurocomputing 36, 171-204, 2001.

[26] RASTOGI, R., PUBLIC: A Decision Tree Classifier that Integrates Building and Pruning. Data Mining and Knowledge Discovery 4: 315–344, 2000.

[26] MARKOVITCH, S., ROSENSTEIN D., Feature Generation Using General Construction Functions, Machine Learning 49: 59-98, 2002.

[28] NIELS, L., MARK HALL, EIBE FRANK, Logistic Model Trees. Machine Learning, 95, 161-205, 2005.

[29] ZHAO, Y., ZHANG Y., Comparison of decision tree methods for finding active objects, Advances in Space Research, 41 (12), 1955-1959, 2008.

[30] GOVERNATORI, G., PASCHKE, A., A New Rule Based Decision Tree Generating Tech., International Symposium, RuleML 2009, Las Vegas, Nevada, USA, 2009.

[31] MICHALSKI, R. S., Synthesis of optimal and quasioptimal variable-valued logic formulas, Proceeding of the 1975 Int. Symposium on Multiple-Valued Logic, Bloomington, Indiana, 76-87, 1975.

[32] CLARK, P., NIBLETT, T., The CN2 Induction Algorithm. Machine Learning, 3(4):261-283, 1989.

[33] COHEN, W. , Fast Effective Rule Induction. In Proceedings of ICML-95, 115-123, 1995.

[34] QUINLAN, J. R., CAMERON-JONES, R. M., FOIL: A midtermreport. In Proc. 1993 European Conf. Machine Learning, pages 3–20, Vienna, Austria, 1993.

[35] AKSOY, M. S., New Algorithms for Machine Learning, Thesis of PhD, University of Walles, Cardiff, United Kingdom, 1993.

[36] TOLUN, M. R., ABU-SOUD S.M., ILA:An Inductive Learning Algorithm For Rule Extraction, Expert Systems With Applications, Vol: 14, p:361-370, 1998.

[37] AKGÖBEK, Ö., ÖZTEMEL, E. Endüktif Öğrenme Algoritmalarının Kural Üretme Yöntemleri ve Performanslarının Karşılaştırılması, SAÜ Fen Bilimleri Enstitüsü Dergisi 10.Cilt, 1.Sayı 2006.

[38] AN, A., CERCONE, N., Rule Quality Measures Improve the Accuracy of Rule Induction: An Experimental Approach, Lecture Notes in Computer Science, Volume 1932, Pages 119-129, 2000.

[39] FURNKRANZ, J., FLACH, P., ROC ‘n’ Rule Learning—Towards a Better Understanding of Covering Algorithms, Machine Learning, Volume 58 (1), 39 – 77, 2005.

[40] LINDGREN, T., Methods for Rule Conflict Resolution, Lecture Notes in Computer Science, Volume 3201, Pages 262 – 273, 2004

[41] MAIMON, O. O., ROKACH, L., Decomposition Methodology For Knowledge Discovery And Data Mining: Theory And Applications (Machine Perception and Artificial Intelligence). World Scientific Publishing Company, July 2005.

[42] NILSSON, N., Introduction to machine learning, Stanford University: Stanford, 2, 1997.

[43] GALLANT, S., I. Neural Network Learning and Expert Systems. MIT Press, Cambridge, Massachusetts., 1994.

[44] KANTARDZIC, M., Data Mining- Concepts, Models, Methods, and Algorithms, Wiley, USA, 45, 2003.

[45] ALPAYDIN, E., Introduction Machine Learning, The MIT Press, Massachusetts, 1-45, 2010.

[46] SIO-IONG, A. O., Data Mining and Applications in Genomics, Springer, UK, 2008.

[47] JONES, M., Artıfıcıal Intellıgence A Systems Approach, Infinity Science Press, New Delhi, 2008.

[48] DANIEL, T. L., Dıscoverıng Knowledge in Data An Introduction to Data Mining, Wiley, New Jersey, 2005.

[49] BERTRAND, C., Principles and Theory for Data Mining and Machine Learning, Springer Newyork , 2009

[50] KANTARDZIC, M., Data Mining- Concepts, Models, Methods, and Algorithms, Wiley, USA, 45, 2003.

[51] WESLEY, C., TSAU Y., Foundations and Advances in Data Mining, Springer Berlin , 2005.

[52] SİLAHTAROĞLU, G., Kavram ve algoritmalarıyla veri madenciliği, Papatya, İstanbul, 10, 2008.

[53] WITTEN, I., FRANK E., Data Mining - Practical Machine Learning Tools and Techniques, Morgan Kaufmann Publishers, San Francisco, 28, 2005.

[54] EVANGELOS, T., Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques, Springer, Louisiana, 2006.

[55] STUART, J., NORVIG, P., Artificial Intelligence A Modern Approach, Pearson, New Jersey, 2003.

[56] ZHANG, D. JEFFREY J. P., Advances in Machine Learning Applications in Software Engineering, Idea Group, London, 2007.

[57] STAHLBOCK, R., CRONE, S. F., Data Mining Special Issue in Annals of Information Systems, Springer, Hamburg, 2010.

[58] KOTSIANTIS, B., Supervised machine learning: classification techniques, Informatica, 31, 249-268, 2007.

[59] AO, S., Data Mining and Applications in Genomics, Springer, UK, 2008.

[60] CHERKASSY, V., MULIER F., Learning from Data - Concepts, Theory and Methods, John Wıley Sons, Inc., Publıcatıon, New Jersey, 340-420, 2007.

[61] WANG, J., Encyclopedia of Data Warehousing and Mining, Information Science reference, Newyork, 2008.

[62] SUMATHI, S., SIVANANDAM, S.N., Introduction to Data Mining and its Applications, Springer-Verlag, Berlin, 2006.

[63] HAN, J., Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, San Francisco, 285, 2006.

[64] RUAN, D., GUOQING, C., KERRE, E., Intelligent Data Mining, Springer, Warsaw, 2005.

[65] MICHALSKI, R.S., Machine Learning - An Artificial Intelligence Approach, Michalski; R.S., Carbonell, J.G. and Mitchell, T.M. (Ed), Morgan Kaufmann, Los Altos, CA, 83-134, 1983.

[66] MITCHELL, T.M., The need for biases in learning generalizations, Readings in Machine Learning, Shavlik, Morgan Kaufmann, San Mateo, CA, 184-191, 1990.

[67] QUINLAN, J.R., Learning efficient classification procedures and their application to chess end games, Machine Learning - An Artificial Intelligence Approach, Eds: Michalski; R.S., Carbonell, J.G. and Mitchell, T.M. (Eds), Tioga Publishing Co, Palo Alto. CA, 463-482, 1983.

[68] DIETTERICH, T.G., Limitations on inductive learning, Proceedings of the 6th International Workshop on Machine Learning (89 ML), Ithaca, NY. and Segre, A.M. (Ed), Morgan Kaufmann, San Mateo, CA, 124-128, 1989.

[69] AKGÖBEK, Ö., Endüktif Öğrenme Algoritmalarının Kural Üretme Yöntemleri ve Performanslarının Karşılaştırılması, SAÜ Fen Bilimleri Enstitüsü Dergisi, 1, 2006.

[70] WANG, L., FU, X., Data Mining with Computational Intelligence, Springer, Singapore, 2005.

[71] YANG, X Dynamic and Advanced Data Mining for Progressing Technological Development, IGI Global, Hershey, 2010.

[72] CHAKRABARTI, S, Data Mining: Know It All, Morgan Kaufmann Publishers, Burlington, 2009.

[73] RUAN, D., GUOQING, C., KERRE, E., Intelligent Data Mining, Springer, Warsaw, 2005.

[74] JENSEN, F., An Introduction to Bayesian Networks. Springer. 1996.

[75] COOPER, G., HERSKOVITS, E.. A Bayesian method for the induction of probabilistic networks from data. Machine Learning, 9, 309-347, 1992.

[76] RISH, I., An Empirical Study of the Naïve Bayes Classifier, In Proceedings of the IJCAI-01 Workshop on Empirical Methods in Artificial Intelligence. 2001.

[77] FRIEDMAN, N., GEIGER, D. GOLDSZMIDT M.. Bayesian network classifiers. Machine Learning 29: 131-163, 1997.

[78] RAANAN, Y., Bayesian Network Structure Learning by Recursive Autonomy Identification, Journal of Machine Learning Research 10 ,1527-1570, 2009.

[79] AHA, D., Lazy Learning. Dordrecht: Kluwer Academic Publishers. 1997.

[80] WETTSCHERECK, D., AHA, D. W. MOHRI, T.. A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. Artificial Intelligence Review 10:1–37, 1997.

[81] MITCHELL, T.. Machine Learning. McGraw Hill, 1997.

[82] ARMENGOL, M., E.. Machine learning from examples: Inductive and Lazy methods. Data Knowledge Engineering 25: 99-123, 1998.

[83] FREUND, Y. SCHAPIRE, R., Large Margin Classification Using the Perceptron Algorithm, Machine Learning 37: 277–296, 1999.

[84] LITTLESTONE, N. WARMUTH, M.. The weighted majority algorithm. Information and Computation 108(2): 212–261, 1994.

[85] NEOCLEOUS, C. SCHIZAS, C.,, Artificial Neural Network Learning: A Comparative Review, LNAI 2308, 300–313, Springer-Verlag Berlin Heidelberg. 2002.

[86] VAPNIK, V., The Nature of Statistical Learning Theory}. Springer Verlag, 1995.

[87] BURGES, C.. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery. 2(2):1-47, 1998.

[88] CRISTIANINI, N. SHAWE-TAYLOR, J.. An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods. Cambridge University Press, Cambridge, 2000.

[89] KENNEDY, R. L., LEE, Y., ROY, B. V, REED, C. D., LİPPMAN, R. P., Solving Data Mining Problems Through Pattern Recognition. Prentice Hall, 1998.

[90] WEISS, S. M., INDURKHYA, N., Predictive Data Mining. Morgan Kaufmann, 1998.

[91] WEISS, S. M., KULIKOWSKI, C. A., Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets,Machine Learning, and Expert Systems.Morgan Kaufmann, 1991.

[92] STONE, M., Cross-validatory choice and assessment of statistical predictions. J. Royal Statistical Society, 36:111–147, 1974.

[93] EFRON, B., TIBSHIRANI, R., An Introduction to the Bootstrap. Chapman Hall, 1993.

[94] KOHAVI, R., A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proc. 14th Joint Int. Conf. Artificial Intelligence (IJCAI'95), vol. 2, pages 1137–1143, Montreal, Canada, Aug. 1995.

[95] FRAKES, W. R., BAEZA Y., Information Retrieval: Data Structures and Algorithms. Prentice Hall, 1992.

[96] WOLPERT, D. H., MACREADY, W.G., No Free Lunch Theorem, Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

[97] QUINLAN, J. R., Bagging, boosting, and C4.5. In Proc. 1996 Nat. Conf. Artificial Intelligence (AAAI'96), volume 1, pages 725–730, Portland, OR, Aug. 1996.

[98] BREIMAN, L., Bagging predictors. Machine Learning, 24:123–140, 1996.

[99] FREUND, Y., SCHAPIRE, R. E., A decision-theoretic generalization of on-line learning and an application to boosting. J. Computer and System Sciences, 55:119–139, 1997.

[100] QUINLAN, J. R., Induction of decision trees, Machine Learning Vol.1, Kluwer Academic Publishers, Boston, 81-106, 1990.

[101] MURTHY, Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey, Data Mining and Knowledge Discovery 2: 345–389, 1998.

[102] BRESLOW, L. A., AHA, D. W., Simplifying decision trees: A survey. Knowledge Engineering Review 12: 1–40, 1997.

[103] ELOMAA, T., The biases of decision tree pruning strategies. Lecture Notes in Computer Science 1642. Springer, 63-74, 1999.

[104] BRUHA, I., From machine learning to knowledge discovery: Survey of reprocessing and postprocessing. , Intelligent Data Analysis, Vol. 4, 363-374, 2000.

[105] QUINLAN, J. R., CAMERON-JONES, R. M.. FOIL: A midtermreport. In Proc. 1993 European Conf. Machine Learning, pages 3–20, Vienna, Austria, 1993.

[106] DAVID, J. C. MACKAY, Information Theory, Inference, and Learning Algorithms, C.U.P, Cambridge, 2003.

[107] Databases For Machine Learning Experiments, http://expdb.cs.kuleuven.be/expdb/index.php, 2010.

[108] The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), http://www.ecmlpkdd2009.net/, 2009.

ÖZGEÇMİŞ

Günay Karlı, 02.09.1972 de İstanbul'da doğdu. İlk ve orta eğitimini Bahçelievler'de tamamladı. 1989 yılında Bağcılar E.M.L Elektronik bölümününden mezun oldu. 1997 yılında Marmara Üniversitesi T.E.F Bilgisayar Bölümünü bitirdi. Yüksek lisansını, 2000 yılında, aynı zamanda araştırma görevlisi olarak çalıştığı Fatih Üniversitesi Bilgisayar Mühendisliği Bölümünde tamamladı. 2000 Yılından sonra, yurt içinde ve dışındaki üniversitelerde öğretim görevlisi olarak çalışan Karlı, 2005 yıından beri Fatih Üniversitesi Bilgisayar Programcılığı Program Başkanı olarak görev yapmaktadır.