ENDÜSTRİYEL OTOMASYON SİSTEMLERİNDE YAPAY ZEKA YÖNTEMLERİ İLE ARIZA TESPİTİ

Oğuzhan ÇÖMLEKCİ

T.C.
BURSA ULUDAĞ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİYEL OTOMASYON SİSTEMLERİNDE YAPAY ZEKA YÖNTEMLERİ İLE ARIZA TESPİTİ

Oğuzhan ÇÖMLEKCİ
(0000-0002-4822-0809)

Dr. Öğr. Üyesi Neyir ÖZCAN SEMERCİ

(Danışman)
(0000-0002-5513-9072)

YÜKSEK LİSANS TEZİ

ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

BURSA – 2020

TEZ ONAYI

Oğuzhan ÇÖMLEKCİ tarafından hazırlanan "ENDÜSTRİYEL OTOMASYON SİSTEMLERİNDE YAPAY ZEKA YÖNTEMLERİ İLE ARIZA TESPİTİ" adlı tez çalışması aşağıdaki jüri tarafından oy birliği ile Bursa Uludağ Üniversitesi Fen Bilimleri Enstitüsü Elektronik Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Danışman : Dr. Öğr. Üyesi Neyir ÖZCAN SEMERCİ

Başkan : Dr. Öğr. Üyesi Neyir ÖZCAN SEMERCİ — İmza
0000-0002-5513-9072
Bursa Uludağ Üniversitesi, Mühendislik Fakültesi,
Elektrik-Elektronik Mühendiliği Bölümü
Devreler ve Sistemler Anabilim Dalı

Üye : Prof. Dr. Güneş YILMAZ — İmza
0000-0001-8972-1952
Bursa Uludağ Üniversitesi, Mühendislik Fakültesi,
Elektrik-Elektronik Mühendiliği Bölümü
Telekomünikasyon Anabilim Dalı

Üye : Dr. Öğr. Üyesi Emel ARSLAN — İmza
0000-0003-4668-392X
İstanbul Üniversitesi-Cerrahpaşa, Müh. Fakültesi,
Bilgisayar Mühendisliği Bölümü
Yazılım Mühendisliği Anabilim Dalı

Yukarıdaki sonucu onaylarım

Prof. Dr. Hüseyin Aksel EREN
Enstitü Müdürü
0000-0003-3908-5139
19.02.2020

Uludağ Üniversitesi Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

12/03/2020

Oğuzhan ÇÖMLEKCİ

ÖZET

Yüksek Lisans Tezi

ENDÜSTRİYEL OTOMASYON SİSTEMLERİNDE YAPAY ZEKA YÖNTEMLERİ İLE ARIZA TESPİTİ

Oğuzhan ÇÖMLEKCİ

Bursa Uludağ Üniversitesi
Fen Bilimleri Enstitüsü
Elektronik Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Neyir ÖZCAN SEMERCİ

Bu tezde; gerçek üretim hatlarından toplanan veri kümelerinin, yapay zeka yöntemleri ile değerlendirilerek otomasyon sistemlerindeki arızaların tespiti üzerine çalışılmıştır. Üretim hatlarından toplanan bu veri kümeleri; Endüstri 4.0 ile birlikte hayatımıza giren "Internet of Things" ve "Big Data" kavramları doğrultusunda elde edilmiş gerçek verilerdir. Elde edilen bu veriler bir yapay sinir ağı algoritması olan perceptron, Rastgele Orman (Random Forest) algoritması ve Gradient Boosting (GBM) algoritmaları kullanılarak işlenmiş ve sistem arızaları sınıflandırılmıştır. Kullanılan yapay zeka yöntemlerinin performansları analiz edilerek seçilen örnek modele en uygun sınıflandırma yöntemi belirlenmiştir. Bu tez çalışması ile yapay zeka yöntemlerinin endüstriyel otomasyon sistemlerinde oluşan arızaların tespitinde başarılı bir şekilde kullanılabileceği gösterilmiştir.

Anahtar Kelimeler: Makine Öğrenmesi, Hata tespiti, Perceptron, Random Forest, Gradient Boosting, Internet of Things, Big Data.
2020, viii + 72 sayfa.

ABSTRACT

MSc Thesis

FAULT DETECTION IN INDUSTRIAL AUTOMATION SYSTEMS WITH ARTIFICIAL INTELLIGENCE METHODS
Oğuzhan ÇÖMLEKCİ

Bursa Uludağ University
Graduate School of Natural and Applied Sciences
Department of Electronics Engineering

Supervisor: Assist. Prof. Dr. Neyir ÖZCAN SEMERCİ

In this thesis; the data sets collected from real production lines are evaluated by using artificial intelligence methods to determine the faults in automation systems.These real datasets collected from production lines are obtained in line with the concepts of "Internet of Things" and "Big Data" which entered our lives with Industry 4.0. These data processed by using Perceptron algorithm (which is an algorithm of Artificial Neural Network), Random Forest algorithm and Gradient Boosting (GBM) algorithms and then the system failures are classified. By analyzing the performances of the artificial intelligence methods, the most appropriate classification method is determined for the selected model. The results of this thesis show that artificial intelligence methods can be used successfully in fault detection for industrial automation systems.

Key words: Machine Learning, Fault Detection, Perceptron, Random Forest, Gradient Boosting, Internet of Things, Big Data.
2020, viii + 72 sayfa.

TEŞEKKÜR

Yüksek Lisans öğrenimim sırasında ve tez çalışmalarım boyunca gösterdiği tüm destek ve yardımlar için değerli hocam Dr. Öğr. Üyesi Neyir ÖZCAN SEMERCİ'ye teşekkürlerimi sunarım.

Her zaman her koşulda yanımda olan sevgili annem, babam ve kardeşime, hayat arkadaşım biricik eşime ve son olarak hayatı yaşamanın en güzel anlamını gözlerinde gördüğüm biricik oğlum Yağız ÇÖMLEKCİ'ye teşekkürlerimi sunarım.

Oğuzhan ÇÖMLEKCİ
12/03/2020

İÇİNDEKİLER

Sayfa

SİMGELER ve KISALTMALAR DİZİNİ

Kısaltmalar	Açıklama
IT	Bilgi İşlem
IOT	Nesnelerin İnterneti
AI	Yapay Zeka
CNN	Hücresel Sinir Ağları
MSSQL	Microsoft Veri Tabanı Uygulaması
ML.NET	Microsoft Makine Öğremesi Kütüphanesi

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

1. GİRİŞ

Günümüz sanayi gelişimine yön veren ve popüler ismiyle Endüstri 4.0 olarak adlandırılan devrim niteliğindeki yeni süreçler, birçok yeni teknolojik gelişmeyi ve araştırma konusunu beraberinde getirmiştir. Endüstrideki sanayi devrimlerine göz atılacak olursa; Birinci sanayi devriminde buharlı motorların icadı ile buhar ve suyun gücü üretimde kullanılmaya başlanmıştır. İkinci sanayi devrimi, elektriğin sanayide kullanılması, elektrikli motorların ve makinelerin üretimiyle gerçekleşmiştir. Üçüncü sanayi devrimi, pek çok kaynakta otomasyon ve IT süreçlerinin üretim teknolojilerinde kullanılması olarak tanımlanır (Yıldız, A . 2018). Son olarak, Endüstri 4.0 olarak tanımlanan süreçte ise robotikleşen sanayi ortamının ve üretim sahasının tamamen dijitalleşme yolunda ilerlediği görülmektedir. IOT süreçleriyle artık üretim sahalarının tüm verilerinin bulut sistemlerinde toplandığı yeni düzenlere geçilmektedir. Bulut sistemlerinde toplanan bu verilerin güvenliği yani siber güvenlik önemli bir konu olduğundan daha çok verilerin lokal bölgelerde toplanması tercih edilmektedir (Tuptuk N, Hailes S. 2018). Endüstriyel ortamlarda tüm üretim makinelerinin ölçülebilen tüm verilerinin dijital ortamlarda olma zorunluluğu, bu yeni düzende veri yığınları ve beraberinde analiz edilecek ve çeşitli çıkarımlar sağlanacak gereksinimler doğurmaktadır. Elimizdeki bu veri yığınlarının gereksinimler doğrultusunda analizleri sonucunda, kendi kendine karar verebilen akıllı fabrikalar olarak tanımlanabilecek yeni bir üretim sürecine, başka bir deyişle sanayi devrimine doğru yol alınmaktadır. Bu süreçte, toplanan verilerden çeşitli tahminler ve tespitler yapılmak amacıyla en yaygın kullanılan yöntemlerden biri makine öğrenmesi dir. Bu noktada sanayi ve üretimde yapay zekadan bahsedildiği çeşitli kaynaklarda da görülmektedir (Jay Lee, Hossein Davari, Jaskaran Singh, Vibhor Pandhare 2018).

Yapay zeka kavramı yüzyıllardır matematikten felsefeye pek çok alanda karşımıza çıkar. Ancak 21. yüzyıl bu kavramın parladığı dönemdir. Düşünen makineler kavramının yaratıcısı Alan Mathison Turing'in II. Dünya savaşında kullanılmak üzere tasarladığı kripto makinesi, bilgisayar ve yapay zeka alanındaki en önemli gelişmelerden biri kabul edilir. II. Dünya savaşı sonrasında Alan Mathison Turing, geliştirdiği 'Turing Makinesi" denen algoritma ile modern bilgisayar kavramının temellerini atmıştır. Yapay zeka alanındaki bir sonraki önemli gelişme ise 1943 yılında Warren S. McCulloch ve Walter

Pitts'in Matematiksel Biyofizik Bülteni'nde yayınladıkları "A logical calculus of the ideas immanent in nervous activit" başlıklı sinir sisteminin içinde olan fikirlerin mantıksal hesabı şeklinde çevrilebilecek çalışması ile yaşanmıştır. Bu çalışmada McCulloch ve Pitts, ilk yapay nöron modelinin teorisini ortaya koymuştur. Yapay zeka kavramının bir çalışma alanına dönüşmesi ise 1955 yılında John McCarthy (Dartmouth College), Marvin Minsky (Harvard Üniversitesi), Nathaniel Rochester (IBM) ve Claude Shannon (ABD) tarafından yaz döneminde gerçekleştirilen araştırma projesiyle başlamaktadır. Bir sene sonra bu araştırmanın sonuçları bir çalıştayda sunulur ve yapay zeka bir çalışma alanı olarak kabul görmeye başlar.

1958 yılında Frank Rosenblatt, Cornell Laboratuvarlarında yapay sinir ağlarının ilkel bir modeli olan Perceptron'u geliştirmiştir. Perceptron bir yazılımdan ziyade bir makine olarak düşünülebilir. Görüntü tanıma için geliştirilmiş bir yapıdır. Ancak görüntü tanımadaki başarısı, eğitilme güçlüğü nedeniyle sınırlı kalmıştır. Bu da bir süreliğine yapay zeka çalışmalarının yavaşlamasına neden olmuştur. Tarihsel süreçler içerisinde çeşitli makine öğrenmesi metotları ile birlikte yapay zeka konusu zaman zaman popülerliliğini arttırmış zaman zaman ise popülerliliğini kaybetmiştir (Michael Haenlein and Andreas Kaplan 2019).

Günümüzde hayatın her noktasına teknoloji ile birlikte taşınan ve kullanımı yaygınlaşan yapay zeka (AI) kavramı, sanayideki dijital dönüşüm sonucunda endüstriyel alanda da son derece popüler hale gelmiştir. Endüstriyel sistemlerden elde edilen çok sayıdaki dijital verinin işlenmesi ve bu veri yığınlarından çıkarımlar yapılması artık bir gerekliliktir. Bu yüksek işlem hacmi nedeniyle tüm bu verilerin yapay bir sistem tarafından değerlendirilmesini ve çıktıların oluşturulmasını bir gereklilik haline getirmiştir (Dopico, M., Gomez, A., De la Fuente, D., García, N., Rosillo, R., Puche, J. 2016). Ayrıca bir çok akademik çalışmaya konu olan yapay zeka yöntemleri, gelişen teknoloji ile birlikte, sanayide kullanılabilecek seviyeye erişmiştir. Endüstriyel kullanım amaçlı olarak yapay zeka yöntemleri üzerine çeşitli kütüphaneler geliştirilmiş, birçok teknoloji devi firma, yapay zeka uygulamaları alanına önemli yatırımlar yapmıştır. Endüstrideki dijital dönüşüm süreci sonucunda yapay zeka yöntemlerinin endüstriyel

üretim, planlama, risk analizi, tahmin, sınıflandırma gibi pek çok alanda yaygın ve etkin olarak kullanılacağı açıktır.

Bu tez çalışması kapsamında, seçilen yapay zeka yöntemleriyle gerçek bir endüstriyel tesiste kullanılmakta olan sıcak hava fırınında (yakıt borularına form vermek amacıyla kullanılan bir otomasyonel makine) oluşan arızaların sınıflandırması yapılmıştır. Sıcak hava fırını, Microsoft Visual Studio platformunda C# form uygulaması üzerinde modellenmiş, oluşabilecek arıza senaryoları kurgulanıp bir veri seti oluşturulmuştur. Gerçek sistemden elde edilen bu veri seti kullanılarak, Microsof Visual Studio platformunda çeşitli yapay zeka yöntemlerinin arıza tespiti için sınıflandırma performansı kıyaslanmıştır. Bu tez çalışması ile oluşturulan veri setinin, ilgili olduğu sıcak hava fırınının dijital dönüşümü sonucunda sisteme otomatik olarak bir yazılım senaryosu ile aktarılabileceği düşünüldüğünde, uygun bir yapay zeka yöntemi ile bu verilerin değerlendirilmesi sonucunda sistem için arıza tespitinin yapılabileceği gösterilmiştir.

2. KURAMSAL TEMELLER

2.1. Yapay Zeka

Zeka, insanın düşünme, akıl yürütme, objektif gerçekleri algılama, yargılama ve sonuç çıkarma yeteneklerinin tamamı olarak tanımlanmaktadır (Türk Dil Kurumu 2020). Yapay Zeka kavramı ise insana özgü bu özelliklerin analiz edilerek makinelere kazandırılması olup insan gibi düşünebilen, yorumlayabilen ve kararlar verebilen algoritmaların ve bilgisayar yazılımlarının geliştirilmesi ile ilgilenmektedir (Balaban Erdal M., Kartal E. 2018).

Yapay zeka hem bilişsel sistemleri simule etmeyi (insan gibi düşünmeyi), hem de "akıllı" sistemler yapılandırmayı (insan gibi davranamyı) kendine amaç edinmiş bir bilimsel disiplindir (Görz, G., Nebel, B. 2002). Yapay zekanın günümüz teknolojisi ile tanımını, "makineleri zeki yapma çalışmaları" olarak görmek mümkündür. Yapay zeka terimi, çeşitli benzetim yollarıyla doğal zekanın yapay hale getirilmesi ve makinelere aktarılmasıdır (Balaban Erdal M., Kartal E. 2018). Makine ifadesiyle mekanik, robotik cisimler akla gelse de günümüzde bilgisayarlar bir makinenin ne olabileceği kavramını oldukça genişletmiştir. Öyle ki işleyen bir bilgisayar sistemi yazılım ve donanımdan meydana gelmekte olup sıklıkla yazılımın kendisi "makine" olarak düşünülmektedir (Nilsson, N. J. 2009).

Özetle yapay zeka; insan gibi düşünen, insan gibi davranan, akılcı düşünen ve akılcı davranan bilgisayar sistemleri ile ilgilenmektedir (Russel, S., Norvig, P. 2003).

Yapay zeka araştırmaları insan beynini model olarak almaktadır. Ancak insan beyni iç organların denetimi, duygular ve hareket gibi birçok görevi aynı anda yapmaktadır. Yapay zeka, beynin sadece zeka kısmı ile ilgilenmektedir. Yapay zeka alanında yapılan çalışmaların temelinde aşağıdaki amaçlar yatmaktadır (Balaban Erdal M., Kartal E. 2018):

- İnsan beyninin işlevlerini bilgisayar modelleri ile anlamaya çalışmak ve insana özgü öğrenme, çıkarım yapma gibi bazı teknikleri incelemek,
- İnsana özgü yetenekleri bilgisayarlara aktarabilmek ve oluşturulan bu sistemleri kullanılabilir hale getirmek,
- Bir alana özgü bilgileri toplayarak bilgi sistemi oluşturmak, genel bilgi sistemleri, yapay zeka iş yardımcıları ve zeki robot ekipleri geliştirmek,
- Yapay zeka alanında yeni araştırmacılar yetiştirmek.

Sadece hayal gücüne dayalı tasarımlar ve bunların sözlü şekilde ifade edilmesi şeklinde başlayan çalışmalar, bilgisayarların ortaya çıkışı, gelişimi ve diğer bilim dallarındaki gelişmelerle birlikte ivme kazanmıştır (Balaban Erdal M., Kartal E. 2018). Günümüzde her bilim dalı alanında yapay zeka çalışmaları yapılmakta, hem yapay zeka yöntemlerinden yararlanılmakta hem de yapay zekanın gelişimine katkı verilmektedir.

2.1.1. Yapay zeka tarihsel gelişim süreçleri

"Yapay Zeka" terimi literatüre ilk kez 1956 yılında John McCarthy tarafından kazandırılsa da (Miller, S. 2011) yapay zeka ile ilgili düşüncelerin ortaya çıkışı antik döneme kadar uzanmaktadır. Örnek olarak Homer, akşam yemeğinde tanrıların önünde bekleyen mekanik hizmetkarlardan bahsetmektedir (Buchanan, B. G. 2005).

Yapay zeka alanında yapılan ve dönüm noktası olarak kabul edilebilecek olaylardan bazıları aşağıda kronolojik olarak verilmiştir (AITopics, 2020) :

- Karel Capek'in "Rossum's Universal Robots" isimli oyunu 1921'de yayınlanmıştır. Bu tarihin özelliği "Robot" kelimesinin ilk kez İngilizcede kullanılışını simgelemektedir. "Robot" terimi "işçi" anlamına gelmektedir ancak 1923'te yapılan İngilizce çeviride kelimenin aslı korunmuştur (Kantrowitz, M. 1994).
- Alan Turing evrensel Turing makinesin 1936-1937 yıllarında tasarlamıştır. Turing makinesi ile modern bilgisayar kavramının temellerini atmıştır.

- Warren McCulloch ve Walter Pitts 1943 yılında sinir ağlarının temelini oluşturan "A Logical Calculus of the Ideas Immanent in Nervous Activity" çalışmasını yayınlamıştır. Bu çalışma bir sinir hücresinin yapay modelinin sunulduğu en önemli çalışmadır. Mcculloch ve Pitts, bu basit sinir hücresinden oluşacak ağların tüm hesap işlemlerini yapabileceğini göstermiştir.
- Turing tarafından 1950 yılında Computing Machinery and Intelligence çalışması yayınlanmıştır. "Turing Testi" olarak bilinen oyunda üç oyuncu mevcuttur. Oyunculardan biri sorduğu sorulara verilen cevaplara bakarak karşısındaki oyuncunun bilgisayar mı yoksa insan mı olduğunu anlamaya çalışır. Bilgisayarın düşünebilen programa sahip olduğu varsayılır. Uygun şartlar sağlandığında Turing'e göre bilgisayarlar bu testi geçebilir.
- John McCarthy "yapay zeka" terimini 1956 yılında Dartmouth Konferansında (yapay zeka için ayrılan ilk konferans) konu başlığı olarak literatüre kazandırmıştır. Alan Newell, J.C. Shaw ve Herbert Simon tarafından yazılmış ve işleyen ilk yapay zeka programı "Logic Theorist" gösterilmiştir. Böylece yapay zeka bir çalışma alanı olarak tüm dünyada kabul görmeye başlar.
- Genel Problem Çözücü (The General Problem Solver-GPS), Newell, Shaw ve Simon tarafından 1957 yılında tanıtılmıştır.
- Arthur Samuel (IBM) dama oyununda bir dünya şampiyonuyla mücadele edebilecek ilk oyun oynayan programı 1952 – 1962 yılları arasında yazmıştır.
- McCarthy (MIT) 1958 yılında LISP dilini geliştirmiştir.
- 1958 yılında Frank Rosenblatt, Cornell Laboratuvarlarında görüntü tanıma için yapay sinir ağlarının ilkel bir modeli olan Perceptron'u geliştirmiştir.
- 1960'ların başında Margaret Masterman ve arkadaşları makine çevirisi için semantik ağlar tasarlamıştır.
- İlk endüstriyel robot firması "Unimation" 1962 yılında kurulmuştur.
- Joseph Weizenbaum (MIT), İngilizce herhangi bir konuda diyalog kuran interaktif program "ELIZA" yı 1965 yılında oluşturmuştur.
- Marvin Minsky ve Seymour Papert basit sinir ağlarının sınırlılığını gösteren "Perceptrons" u 1968 yılında yayınlamıştır.
- 1969 yılında hareketlilik, algı ve problem çözmeyi birleştiren "Stanford Research Institue" robotu "Shakey" tanıtılmıştır.

- 1980'lerin ortalarında sinir ağları, geri yayılım (backpropagation) algoritması ile birlikte geniş ölçüde kullanılır hale gelmiştir.
- Hücresel sinir ağı modeli ve teorisi ilk defa 1988 yılında, Chua ve Yang tarafından ortaya atılmıştır. IEEE Transactions on circuits and systems dergisinde, "Cellular neural networks: theory" başlığı ile yayınlanan bu makale hücresel sinir ağlarının (Cellular Neural Networks, CNNs) model ve teorisinin temelini oluşturmuştur (Chua L.O., Yang L., 1988). Hücresel sinir ağları yapısı ve paralel sinyal işleme yeteneği sayesinde gerçek zamanlı sinyal işleme alanında yaygın olarak kullanılmaya başlanmıştır.
- 1990'lı yıllarda makine öğrenmesi, akıllı ders, durum-tabanlı akıl yürütme, çoklu-ajan planlama, zamanlama, belirsiz akıl yürütme, veri madenciliği, doğal dil anlama ve çeviri, görüntü, sanal gerçeklik, oyunlar ve diğer konularda anlamlı gösterimler ile yapay zekanın tüm alanlarında büyük gelişmeler sağlanmıştır.
- Satranç programı "Deep Blue" satranç dünya şampiyonu Garry Kasparov'u 1997 yılında yenmiştir.
- 1990'lı yılların sonunda yaygın internet kullanımında web örümceği (crawler) ve diğer yapay zeka tabanlı bilgi çıkarımı programlarının kullanımı gerekli hale gelmiştir.
- 1993 yılında çok fonksiyonlu makine (CNN-UM) mimarisi Roska T. ve Chua L.O. tarafından geliştirilmiştir (Roska T., Chua L.O., 1993). Bu makine hem analog hem de lojik bir mimariye sahiptir ve program depolayabilmektedir.
- 1994 yılında ilk CNN evrensel yongası geliştirilmiş ve tüm dünyaya duyurulmuştur (Dominguez-Castro R., Espejo S., Rodriguez-Vazquez A., Carmona R., 1994). Zaman içerisinde ACE4k (Linan G., Espejo S., Dominguez-Castro R., Rodrigez-Vazquez A., 2002), ACE16k (Rodriguez-Vazquez A., 2004) ve EYE-RIS gibi pek çok CNN çok fonksiyonlu makine yongası geliştirilmiştir.
- 2000'lerde çok sayıda ve çeşitte ticari akıllı oyuncaklar geliştirilmiştir.
- Cyntha Breazeal, duygularını bir yüz ile ifade edebilen robot "KISMET" i anlatan, "Sosyal Makineler" hakkındaki tezini yayınlamıştır.
- 2005'te Stanford'ın otonom aracı Stanley, "DARPA Grand Challenge" yarışını kazanmıştır.

- 2005’den günümüze yapay zeka alanındaki çalışmalar devam etmekte olup gündelik hayatta kullanılan akıllı telefonlardan oyunlara, sanayide birçok robotik teknolojiyle, üretimden üç boyutlu yazıcılara kadar birçok alanda kullanımı giderek yaygınlaşmaktadır.

2.1.2. Yapay zeka teknolojileri

1950’li ve 1960’lı yılların başlarında yapay zeka çalışmalarının öncüleri bulmaca çözme, satranç ve dama oynama, teoremlerini ispat etme, basit sorulara cevap verme ve görüntü sınıflandırma gibi problemler ile ilgilenirken, bugünün yapay zeka programları insan bilişsel yeteneklerinin bazılarına benzeyebilmektedir ve hatta insanın yapabildiğinden daha iyisini yapabilmektedir (Nilsson, N. J. 2009).

Bunların yanı sıra yapay zeka teknolojileri problem çözümleme, oyunların modellenmesi, bilgilerin modellenmesi, otomatik teorem ispatı, uzman sistemler, doğal dilin işlenmesi, örüntü tanıma, robotik ve bunun gibi daha bir çok alanda gelişmeler sağlamıştır (Nabiyev, V. V. 2005).

Bugün yapay zeka çalışmalarının ürünlerine günlük hayatta farklı şekillerde rastlamak mümkündür. Ev aletleri, gelişmiş sürücü destek sistemleri, haritadan yol bulma, web sitelerinde kullanılan ziyaretçilerin sevebileceği ürünleri öneren sistemler, insanların simule edilen bir dünyada yer alan sanal bir karakterle etkileşebildiği bilgisayar oyunları gibi uygulamalar yapay zekanın nasıl kullanıldığına dair örneklerdir (Nilsson, N. J. 2009).

Yapay zeka çalışmalarının sonucunda çeşitli teknolojiler doğmuştur. Günümüzde altmıştan fazla yapay zeka teknolojsinin olduğu söylenmektedir (Öztemel E. 2006). Yapay zeka teknolojilerinden uzman sistemler, genetik algoritmalar ve bulanık mantığın yanında insanlardaki biyolojik sinir hücresinin çalışma prensibine dayanan yapay sinir ağlarını da içine alan “makine öğrenmesi” bu teknolojilere ek olarak yapay zekanın dördüncü evresini oluşturmaktadır (Özen, Z., Kartal, E., Gülseçen, S. 2017).

Yapay zeka tekniklerinin en popülerleri arasında makine öğrenmesi bulunmaktadır. Yapay zekanın alt dalları ve kesişimleri çok net olmamakla birlikte, aşağıda genel kabul gören başlıklar verilmiştir. Bu ana başlıklar pek çok alt başlık içeririr:

- Uzman Sistemler
- Bulanık Mantık
- Doğal Dil İşleme
- Yapay Sinir Ağları
- Genetik Algoritmalar
- Örüntü Tanıma
- Makine Öğrenmesi

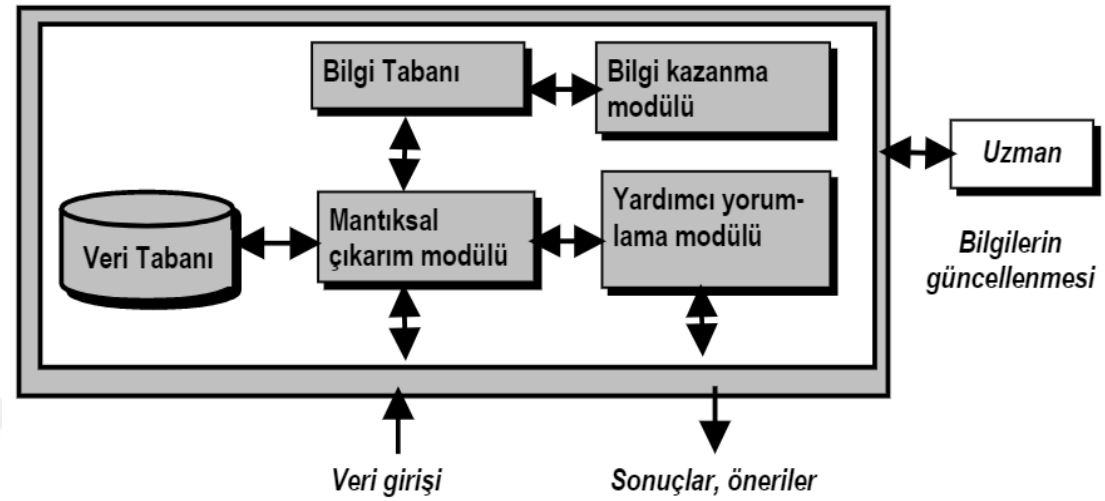
2.1.3. Uzman sistemler

Uzman sistemler insanların uzman oldukları bir konudaki kıyaslama, karşılaştırma ve bunların sonucunda karar verme yeteneğini modelleyen programlardır (Nabiyev V.V.. 2005). Uzman sistemlerin getirdiği kolaylıklardan bir tanesi kurumlar için uzmanlık gerektiren bir işle ilgili personel bulmadaki zaman ve finansal kısıtları ortadan kaldırmasıdır. Aynı zamanda uzman personellere sahip işletmeler bu personellerin işten ayrılması sonucu uzmanlık bilgilerini de beraberinde götürmesinden kaynaklı problemler yaşayabilmektedir. Uzman sistemler, bir işletmede bir işin yapılması, bir kararın alınması, bir problemin çözülmesi sırasında sahip olduğu bilgiyi kullanarak çözüm sağlamaktadır (Balaban Erdal M., Kartal E. 2018).

Uzman sistemlerin en önemli özelliği veri tabanını yeni programa ihtiyaç duymaksızın sürekli genişletebilmesi ve kullanıcı hatalarını düzeltebilme kabiliyetleridir. Bir uzman sistem kullanıcısına bir problem karşısında uzman deneyimi ve yetkinliğinde cevap üretebilir.

Uzman sistemlerin bileşenlerinde yer alan kullanıcı ara yüzü; bilgi kazanma, bilgi tabanı ile hata ayıklama, test durumlarını çalıştırma, özet sonuçlar üretme gibi görevleri yerine getirmektedir (Şekil 2.1.). Bir diğer bileşen olan çıkarım mekanizması ise arama ve

çıkarımın yer aldığı kısımdır. Uygun bilgi için bilgi tabanını taramakta ve mevcut problem verisine dayanarak çıkarımda bulunmaktadır (Balaban Erdal M., Kartal E. 2018).



Şekil 2.1. Bir uzman sistemin genel yapısı (Nabiyev V.V., 2016)

Uzman sistemler sayesinde uzmanlık bilgi ve tecrübesi bilgi tabanında saklanarak istenilen zamanda çıkarımlar hızlı ve kolaylıkla elde edilebilir. Böylece işletmelere finansal ve zamansal kazanç sağlanır.

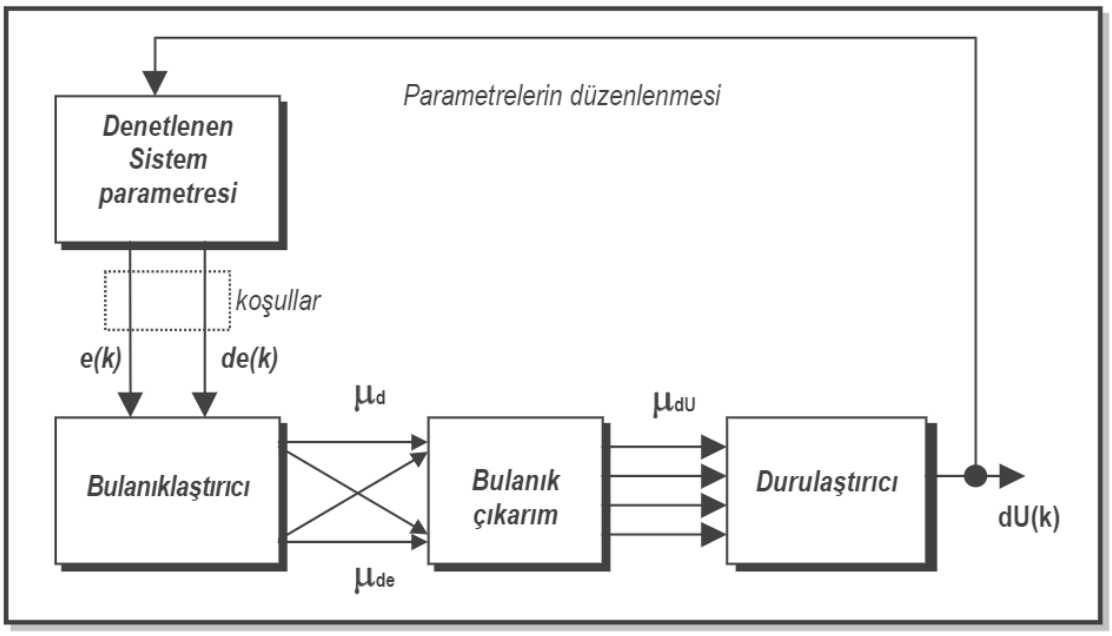
2.1.4. Bulanık Mantık

Bulanık mantığın temelleri 1961 yılında Lütfü Aliasker Zade tarafından oluşturulmuştur. Mantığı dereceli üyelik kavramına dayanır. Belirsiz ve kesin olmayan verileri modellemede çok başarılıdır. Klasik Aristo mantığında önermeler "doğru" ya da "yanlış" olarak değerlendirilirken, bulanık mantık önermelerinin modellenmesinde doğru ve yanlış arasındaki değerler kullanılır. Klasik kümeleme mantığında bir kümeye ait olup olmama durumu sırasıyla 1 ve 0 olarak tanımlanacak olursa, bulanık mantık için bir kümeye ait olma durumu bir üyelik derecesiyle tanımlanır. Bu nedenle bulanık mantık gerçek olayların modellenmesinde oldukça başarılı yaklaşımlar sunar.

Bulanık mantığa dayalı sistemlerin çalışma yapısı aşağıdaki gibidir (Sarı, M., Murat, Y., Kırbalı, M. 2005):

- Ele alınan problem tanımlanır ve probleme uygun parametreler seçilerek üyelik fonksiyonları oluşturulur. Üyelik fonksiyonları evrensel kümenin elemanına [0 1] aralığında bir üyelik derecesi belirler. Böylece girişler bulanıklaştırılır.
- Kural tabanlı bulanık sistemler EĞER-OHALDE kurallar dizisiyle tanımlanır. Kural tabanı, problemdeki ilgili parametrelere ve bulanık alt kümelere göre oluşturulan kurallar dizisidir. Bulanıklaştırılan girişler bu kural setiyle işlenir.
- Kural setiyle işlenen girişlerden elde edilen çıkarımlar bulanık ya da gerçek olabilir. Çıkarım bulanık ise sistem durulaştırma işlemine ihtiyaç duyar.
- Bulanık olan değerin yeniden durulaştırılması, bir başka ifade ile sayısal değerlere dönüştürülmesi yöntemi belirlenir.

Şekil 2.2. de bir bulanık denetleyicinin genel yapısı gösterilmiştir.



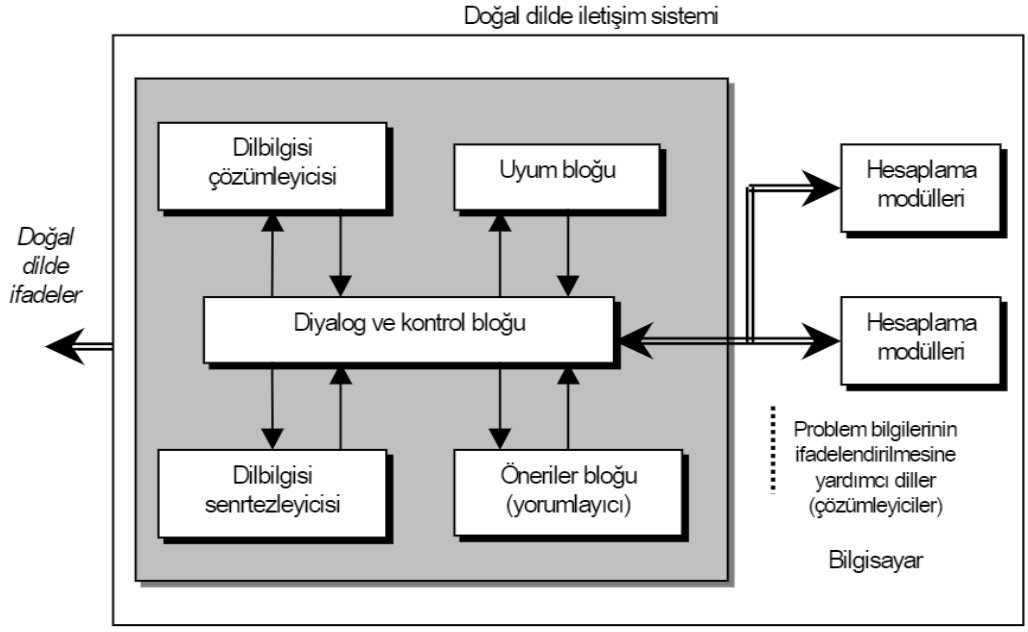
Şekil 2.2. Bulanık denetleyicinin genel yapısı (Nabiyev V.V., 2016)

2.1.5. Doğal dil işleme

Vasif V. Nabiyev doğal dil işlemeyi Yapay Zeka adlı kitabında " *Doğal dil işleme ana işlevi bir doğal dili çözümleme, anlama, yorumlama ve üretme olan bilgisayar sistemlerinin tasarımını ve gerçekleştirmesini konu alan bir mühendislik alanıdır."* cümlesiyle tanımlamıştır (Nabiyev V.V., 2005).

Bilgisayarlı çeviri sistemlerinin ilk uygulmalarından sonra doğal dilin çözümlenmesinin zorlukları da ortaya çıkmıştır. 60'lı yıllar ile birlikte doğal dile yakın olacak kullanıcı ara birimleri geliştirilmeye başlanmıştır. Bu sistemlerde yalnızca metnin dili değil aynı zamanda da kullanım psikolojisinin etkileri de göz önünde bulundurularak çalışmalar yapılmıştır (Nabiyev V.V., 2016).

Geliştirilmeye çalışılan kullanıcı ara birim sistemlerinde mönülü, anket biçimli sorgulamalı yapılar kullnılmıştır. Bu sistemlerde değişmez sabit biçimli yapılar olduğundan dolayı kullanıcının kısıtlandığı durumlar oluşmaktadır (Şekil 2.3.). Daha çevik ve kullanıcının kısıtlarının azaltılacağı sistem doğal dil işleme sistemleri yardımıyla gerçekleştirilebilmektedir (Nabiyev V.V., 2016).

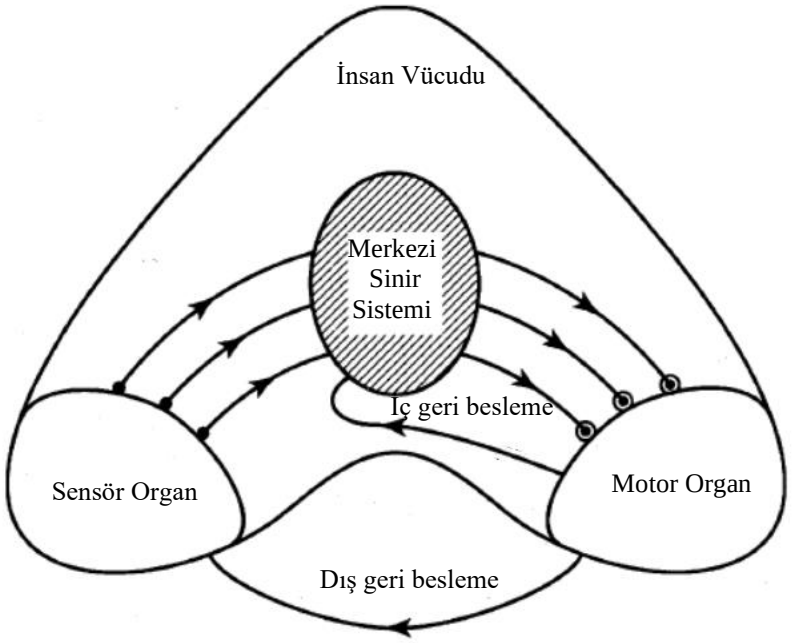


Şekil 2.3. Doğal dil işleme ara birim yapısı (Nabiyev V.V., 2016)

2.1.6. Yapay sinir ağları

İnsan beynini üstün kılan idrak etme yeteneğinin bilgisayarlara aktarılması amacıyla yapılan yapay zeka çalışmaları alt dallarından biri olan "yapay sinir ağları" (YSA), insan beyninin sinir ağlarını taklit eden bilgisayar programlarıdır.

Yapay sinir ağı modelleri için örnek alınan yapı insan sinir sisteminin gelen bilgiyi değerlendiren, karşılaştıran, işleten ve gerektiğinde cevap üreten merkezi sinir sistemidir. Sinir sistemindeki en küçük birim olan biyolojik nöron esas alınarak yapay sinir ağları için nöron modelleri tasarlanmıştır (Şekil 2.4.).



Şekil 2.4. Sinir sisteminde bilgi akışı (Zurada J. M., 1992)

Yapay sinir hücreleri bir araya gelerek yapay sinir ağlarını oluşturur. Yapay sinir ağları giriş katmanı, gizli katman ve çıkış katmanı olmak üzere genellikle üç katmandan oluşur. Gizli katman içerisindeki katman sayısı birden fazla olabilir. Bunun için bir kural yoktur. Uygulamanın türüne ya da deneme-yanılma yöntemiyle, ağın performansı gözlenerek gizli katman sayısına karar verilir.

Yapay sinir ağları üzerine araştırmalar günümüzde iki alan üzerinde yoğunlaşmıştır. Bunlar;

- İleri Beslemeli Çok Katmanlı Ağlar
- Geri Beslemeli Ağlar

İleri beslemeli ağlar genellikle örüntü tanıma problemlerinde sıklıkla kullanılırken geri beslemeli ağlar ise optimizasyon problemlerinin çözümünde tercih edilmektedir.

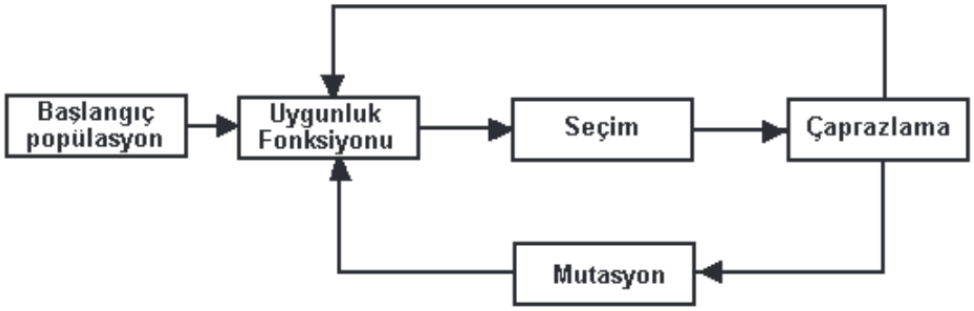
1943 yılında McCulloh-Pitts modeli ile başlayan ve insan beyninin merkezi sinir sisteminin görevlerini modelleme çalışmaları 1980'li yıllarda hız kazanmıştır. Özellikle paralel bilgi işleme yeteneği sebebiyle işlemlerde sağladığı hız, hatalı ya da eksik bilgilere karşı toleransı uygulamalarda tercih edilmesine sebep olmuştur.

2.1.7. Genetik algoritmalar

Genetik algoritmalar evrimsel yaklaşım ilkelerini baz alarak, rastlantısal araştırma yöntemlerini kullanarak kendi kendine öğrenme ve karar verme sistemlerinin düzenlemesini amaçlar. Genetik algoritmaların çalışma prensipleri aşağıda açıklanmıştır (Bolat, B., Erol, K. O., İmrak, C. E.):

- Problemin olası çözümlerinin kodlandığı bir çözüm grubu oluşturulur. Biyolojideki benzerliğinden dolayı bu çözüm grubuna popülasyon, çözümlerin kodlarına da kromozom adı verilmektedir. Problem için n kromozomlu bir popülasyon ele alınır.
- Her bir kromozom için, kromozomların ne kadar iyi olduğunu bulan uygunluk fonksiyonu hesaplanır.
- Yeni bir popülasyon oluşuncaya kadar seçim, çaprazlama ve mutasyon adımları izlenir.
- Yeni popülasyon kabul edildikten sonra eskileri ile yer değiştirilir ve hedeflenen uygunluk değerine ulaşıldığında program durdurulur. Popülasyondaki en iyi çözüm alınır.

Şekil 2.5. de genetik algoritmalarda evrimleşme döngüleri gösterilmiştir.



Şekil 2.5. Genetik algoritmalarda evrimleşme döngüleri (Nabiyev V.V., 2016)

2.1.8. Örüntü tanıma

Vasif V. Nabiyev, örüntü tanımayı Yapay Zeka adlı kitabında " *Görüntü olayında, nesnelerin ne olduklarının belirlenmesine örüntü tanıma süreci ya da kısaca tanıma denir."* cümlesiyle tanımlamıştır (Nabiyev V.V., 2016).

Bilgisayar tabanlı sistemlerdeki en önemli işlemlerden biri kullanıcı tanımlamadır. Kullanıcı tanıma işlemleriyle birlikte sağlanacak kullanıcı girişi güvenliğinde verilen şifre, anahtar gibi değerlerin unutulabilmesi, tahmin edilebilmesi gibi çeşitli dezavantajlar mevcuttur. Bu dezavantajları biyometrik sistemler büyük oranda ortadan kaldırmaktadır. Biyometrik sistemlerde kullanılan parametrelere örnek olarak parmak izi, retina, el ve yüz yapısı, DNA verilebilir (Nabiyev V.V., 2016).

Yapay zekanın örüntü tanıma yöntemleriyle birlikte biyometrik özellikler kullanılarak çeşitli güvenlik sistemleri tasarlanabilmektedir. Günümüzde biyometrik sistemler bankacılıktan kriminolijiye, endüstride yetkin kişinin makineyi çalıştırabilmesinden güvenlik amaçlı geçiş sistemlerinde giriş çıkış işlemlerinin yapılmasına kadar pek çok noktada kullanılmaktadır (Nabiyev V.V., 2016).

Örüntü tanıma yöntemlerinin en popülerlerinden birisi olan yüz tanıma, bugün günlük hayatta akıllı telefonlarımızdaki kilit ekranlarının açılma işlemlerinde dahi kullanılmaktadır. Yüz tanıma sistemleri oldukça geniş bir kullanım alanına sahiptir. Şekil 2.6. da yüz tanımanın aşamaları gösterilmiştir.

Şekil 2.6. Yüz tanımanın aşamaları (Nabiyev V.V., 2016)

2.1.9. Makine öğrenmesi

Makine öğrenmesinin literatürde kabul görmüş en iyi tanımlarından biri Mitchell tarafından yapmıştır. Mitchell makine öğrenmesini, "*Eğer bir bilgisayar programının "G" görevlerinde "P" ile ölçülen performansı, deneyim "D" ile artıyorsa, o bilgisayar programının bazı G görevlerinin sınıflarına ve performans ölçüsü P'ye göre deneyim D'den öğrendiği söylenmektedir.*" biçiminde açıklamıştır (Mitchell, T. M. 1997).

Makinenin, kendisine deneyim olarak verilen veri setini kullanarak öğrenmeyi sağlayabilmesi, bu amaçla geliştirilmiş ve problem tipine uygun çeşitli algoritmalar yardımı ile sağlanmaktadır. Sonuçta elde edilen performansa göre makinenin öğrenip öğrenmediği ortaya çıkmaktadır. Eğer performans yüksekse, makinenin doğru bir çıktı üretmesi mümkün olacaktır. Performansın düşük olması durumunda ise makine öğrenmesini etkileyen faktörlerde değişiklik yapılarak performansın optimize edilmesi gerekir. (Balaban Erdal M., Kartal E. 2018).

Makine öğrenmesini etkileyen faktörlerden ilki makineye deneyim olarak sunulan veri setidir. Tıpkı insanlarda olduğu gibi deneyim ne kadar fazla ve farklı olası durumları içeriyorsa, öğrenmeye o kadar pozitif yönde katkı sağlayacaktır (Balaban Erdal M., Kartal E. 2018).

İkinci faktör, veri setinde sonuca etkisi olduğu düşünülen değişkenlerin bulundurulmasıdır. Örnek olarak bir tenis maçının oynanıp oynanmayacağını tahmin edebilmek için havanın ısı, nem veya rüzgar gibi özelliklerine bakılırken, bir hastanın ölüm riskinin belirlenmesi için yaş, cinsiyet, sigara/alkol kullanıp kullanmadığı gibi bilgiler birer özellik olarak dikkate alınmaktadır. Makine öğrenmesi çalışmalarında bu özellikler nitelik (attribute) olarak adlandırılmaktadır. Bazı problemlerde, birbiriyle yüksek ilişkili ya da çözüme katkısının çok az olduğu tespit edilen nitelikler elenebilmektedir (Balaban Erdal M., Kartal E. 2018).

Üçüncü faktör, seçilen öğrenme stratejisidir. Öğrenme stratejisi, hem makinenin öğrenmesi istenilen görevle hem de bunun için gerekli veri setiyle yakından ilişkilidir. Çünkü makine öğrenmesi bazı problemlerin çözümü için veri setinde çıktı değerlerine ihtiyaç duyarken, bazılarında ise bu değerlere ihtiyaç duymaz. Çıktı değeri, çalışmalarda hedef nitelik (target attribute) olarak adlandırılmaktadır. Hedef niteliğe ihtiyaç duyulan problemlerin daha çok tahmin ve sınıflandırma problemleri olduğu görülmektedir. Dolayısıyla makine öğrenmesinde ele alınacak probleme göre öğrenme stratejisinin belirlenmesi ve yine bu stratejiye uygun veri setinin temin edilmesi önemlidir (Balaban Erdal M., Kartal E. 2018).

Dördüncü faktör ise öğrenme için kullanılan algoritma ve varsa algoritmaya ait parametrelerdir. Algoritma, verilen bir problemin çözümü için adım adım izlenmesi gereken yol şeklinde ifade edilebilir. Makinenin de elbette sahip olduğu veri setini göz önünde bulundurarak öğrenebilmesi için gereken birtakım adımlar mevcuttur. Literatürde bu amaçla geliştirilmiş olan pek çok algoritma mevcuttur. Makine öğrenmesi için veri seti ve algoritma temelinde model ya da modeller tasarlanmakta, en yüksek performansın elde edildiği modelin kullanılması tercih edilmektedir (Balaban Erdal M., Kartal E. 2018).

Tüm modellerin yer aldığı küme M = $\{M_1, M_2, \ldots, M_n\}$ ile gösterilsin. M kümesinde aynı öğrenme modeli farklı biçimlerde denenebileceği gibi, farklı öğrenme modelleri de bir arada kullanılabilmektedir (Rai, P. 2011). Örnek olarak;

- M_1 : k – En Yakın Komşu Algoritması (k = 5)

- M_2 : k – En Yakın Komşu Algoritması (k = 9)
- M_3 : Naive Bayes Sınıflandırıcı
- M_4 : Karar Ağaçları
- M_5 : Perceptron
- M_6 : Yapay Sinir Ağları

Makine öğrenmesi için geliştirilen algoritmalar bilgisayarlara kolaylıkla aktarılabildiğinden makine, basitçe günlük hayatta sıklıkla kullandığımız bilgisayarlar olarak düşünülebilinir. Performansı yüksek bir modelin bilgisayar programları ile farklı donanımlara aktarılması kolaylıkla sağlanmaktadır. Makine öğrenmesi alanı aşağıda verilen üç temel araştırma üzerine kurulmuştur (Camastra, F., Vinciarelli, A. 2008) :

- Göreve yönelik çalışmalar ; örneğin, daha önceden tanımlanmış görevlerde performansı arttırmak için öğrenme sistemlerinin geliştirilmesi,
- Bilişsel simülasyon, diğer bir deyişle insana ait öğrenme süreçlerinin incelenmesi ve bilgisayar simülasyonunun yapılması,
- Teorik analizler; örneğin, pratik alandan bağımsız biçimde, mevcut öğrenme metotları ve algoritmalarının teorik incelenmesi.

2.2. Makine Öğrenmesi Stratejileri

Öğrenme, belli bilgi, beceri ve anlayışlar edinme biçiminde tanımlanmaktadır. İnsanlar, çocukken anne ve babasından yemek yeme, konuşma, yürüme gibi temel becerileri, okulda ise öğretmenlerinden dil, tarih, matematik, fizik vb. birçok dersi öğrenirken, okul hayatı sonrasında ise kendi kendine öğrenme sürecini devam ettirmektedir. Bilgisayar bilimleri, öğrenme alanında canlılar üzerinde yapılan çalışmaları, makineler üzerine aktarmıştır (Balaban Erdal M., Kartal E. 2018).

Kendi kendine öğrenebilen, veriden çıkarım yapabilen makineleri tasarlama ve geliştirme düşüncesi, gelecekte zeki robotların aramızda dolaşabileceğine işaret etmektedir. Nitekim son yıllarda düzenlenen konferanslar ve çalışma grupları bu düşünceyi destekler niteliktedir. İnsan beyninin ve sinir sisteminin karmaşıklığı ve tamamen keşfedilmemiş olması, bu yapının aynısının makinelere aktarılabilmesini güçleştirmektedir. Ancak

yapıyı tamamen kopyalamak yerine, bu sistemlerden esinlenme veya bu sistemlerin çalışma biçimlerine benzetim yöntemleri ile yapay sinir ağları, genetik algoritmalar gibi bazı teknikler geliştirilmiştir (Balaban Erdal M., Kartal E. 2018).

Makine öğrenmesi tekniklerinin gruplandırılmasında literatürde farklı yaklaşımlar bulunmaktadır. Carbonell ve arkadaşları makine öğrenmesi sistemlerinin sınıflandırılmasını üç farklı grupta incelemiştir (Carbonell, J. G., Michalski, R. S., Mitchell, T. M. 1983):

- Kullanılan öğrenme stratejisi: Alışkanlıklarından öğrenme ve yeni bilginin doğrudan nakledilmesi, talimatlardan öğrenme, benzeterek öğrenme, örneklerden öğrenme, gözlem ve keşif yolu ile öğrenme,
- Öğrenen tarafından edinilen bilginin (karar ağaçları, biçimsel gramer, üretim kuralları vb.) ya da becerinin gösterimi,
- Bilgi edinilen performans sisteminin uygulama alanı (ziraat, kimya, bilişsel modelleme, matemeatik, uzman sistemler vb.)

Öğrenme yöntemleri, danışmanlı öğrenme (supervised learning), danışmansız öğrenme (unsupervised learning) ve pekiştirmeli öğrenme (reinforcement learning) biçiminde üç grupta incelenebilmektedir (Bishop, Christopher M. 2007) (Camastra, F., Vinciarelli, A. 2008) (Balaban Erdal M., Kartal E. 2018). Son zamanlarda, çeşitli araştırma grupları, bir dizi yaygın danışmanlı, yarı danışmanlı, danışmansız ve pekiştirmeli öğrenme problemleri konuları üzerinde birleşmiştir (Alpaydın, E. 2009).

2.2.1. Danışmanlı öğrenme

Danışmanlı öğrenmede amaç, bir dizi girdi değerine dayanarak, çıktı değerinin tahmin edilmesidir (Hastie Trevor, Tibshirani Rober, Friedman Jerome 2008). Danışmanlı öğrenme probleminde, eğtim örneği (training sample) ya da eğitim kümesi (training set) olarak da bilinen, verilerin girdi-çıktı ikililerinde; görev, hataları olabildiğince en aza indirerek, gelecekteki girdi-çıktı gözlemlerini tahmin edebilecek, her girdiyi bir çıktıya haritalayan deterministik bir fonksiyon bulabilmektir (Camastra, F., Vinciarelli, A. 2008).

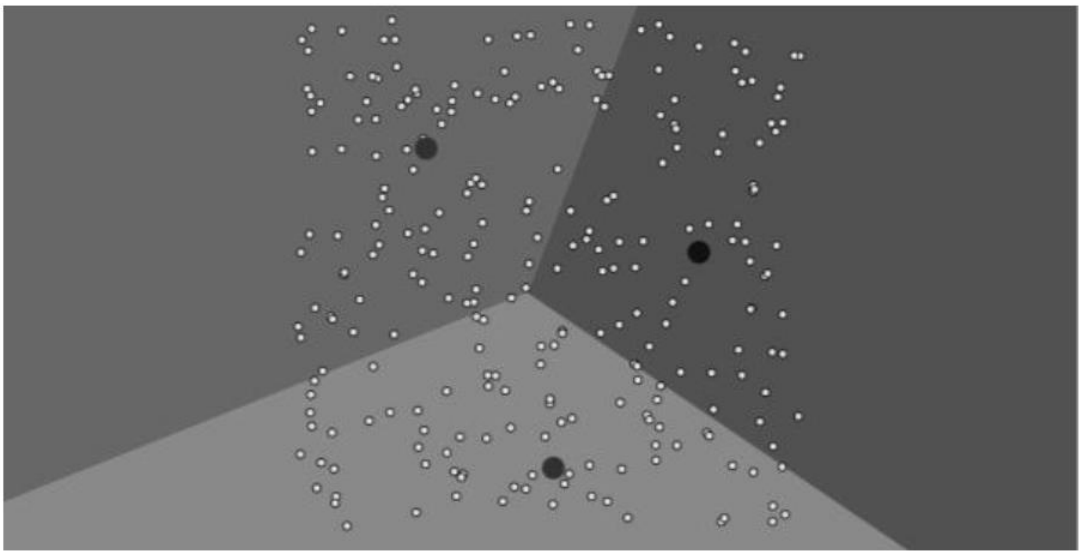
Eğitim verisinde, girdi vektörleri ve onlara ilişkin hedef vektör örneklerinden oluşan uygulamalar danışmanlı öğrenme problemleri olarak bilinmektedir (Bishop, Christopher M. 2007). Bu öğrenmede elde edilecek sonuç, tahmini istenen çıktı değerinin ya da etiketlenmiş değerin bilinmesi durumudur. Diğer bir deyişle, girdi değerleri bir danışman (veya eğitmen) tarafından çıktı değerlerine etiketlenmektedir. Bu koşullarda danışman desteğinde eğitilen veri seti çıktı değerlerinin tahmin edildiği bir model oluşturulur. Bu model, tahmin edilen çıktı değerleri ile bilinen çıktı değerleri arasındaki farkın en aza indirildiği bi modeldir (Balaban Erdal M., Kartal E. 2018).

Amacı her girdi vektörüne sonlu sayıdaki ayrık kategorilerden birini atama olan durumlar, "sınıflandırma (classification)" problemi olarak tanımlanmaktadır (Bishop, Christopher M. 2007). Çıktı uzayındaki her elemente sınıf (class), sınıflandırma problemini çözen öğrenme algoritmasına da sınıflandırıcı (classifier) adı verilmektedir (Camastra, F., Vinciarelli, A. 2008). Eğer istenen çıktı bir ya da daha fazla sürekli değerden oluşmakta ise bu yönteme regresyon (regression) denilmektedir (Bishop, Christopher M. 2007).

Danışmanlı öğrenme algoritmaları geçmiş deneyimlerden faydalanarak yeniler için çıkarım yapabilme yeteneğine sahiptir; ancak en iyi sınıflandırmayı yapabilmek probleme uygun algoritmanın seçilmesini de gerektirmektedir (Balaban Erdal M., Kartal E. 2018).

2.2.2. Danışmansız öğrenme

Danışmansız öğrenmede, çıktı değeri olmaksızın amaç girdi değerleri arasındaki ilişki ve örüntüleri tanımlayabilmektir (Hastie Trevor, Tibshirani Rober, Friedman Jerome 2008). Danışmansız öğrenmede, çıktı değerinin olmadığı veya bilinemediği durumlarda sadece girdi değerlerinden hareket edilerek girdilerin gruplandırılması, diğer bir deyişle kümelenmesi (clustering) sağlanmaktadır (Şekil 2.7.). Danışmanlı öğrenmede sınıflandırma olarak ele alınan algoritmalar, danışmansız öğrenmede öğrenmeye yardımcı bir danışman söz konusu olmadığı için verilerin yoğunlukları veya benzerlikleri dikkate alınarak gruplandırılmaktadır. Dolayısıyla danışmansız öğrenme algoritmaları kümeleme algoritmaları olarak da tanımlanabilmektedir (Balaban Erdal M., Kartal E. 2018).



Şekil 2.7. Bir k ortalama kümeleme örneği (Gürsakal, N. 2017)

Danışmansız öğrenme problemlerinde amaç;

- Veri içinde benzer örneklerin gruplarını keşfedebilmek – kümeleme,
- Girdi uzayında verinin dağılımını belirleyebilmek – yoğunluk tahmini ya da
- görselleştirme amacı ile çok boyutlu uzayı, iki ya da üç boyutlu uzaya yansıtabilmektir (Bishop, Christopher M. 2007).

2.2.3. Pekiştirmeli öğrenme

Pekiştirmeli öğrenme yaklaşımı, öğrenen makinenin davranışını, dinamik bir ortamda deneme – yanılma şeklinde bir etkileşime girerek öğrenmesidir. Bunun için arzu edilen doğruluk değerine sahip bir etiket bilgisine gereksinim yoktur ve geçici etiketlerle öğrenme süreci başlatılmaktadır. Tek gereken bu geçici etiketlerin doğru ya da yanlış olduğunu belirten öğretici bir geri bildirimdir. Burada geri bildirim, bir şeyin yanlış olduğunu ya da doğru olduğunu belirten fakat neden yanlış olduğunu söylemeyen bir eleştirmen gibidir (Duda, R. O., Hart, P. E., Stork, D. G. 2001). Bu yaklaşımda öğrenmeyi destekleyen bir danışman vardır. Burada gerçek değer sunmak yerine danışman tarafından çıktı değerinin ne derece doğru olduğunu belirten bir sinyal, skor veya derece bildirilmektedir.

2.3. Arıza Sınıflandırmasına Yönelik Temel Makine Öğrenmesi Algoritmaları

Bir bilgisayar programının yazılmadan önce programa ait algoritmanın yazılmasına ihtiyaç duyulduğu gibi; kendi kendine karar verebilen, tahminde bulunan bir programın geliştirilmesinde de yine algoritmalardan faydalanılmaktadır. Algoritmalar, ele alınan problemin çözümü için kurulan modelde etkin rol oynamaktadır. Literatürde, makine öğrenmesi için geliştirilmiş olan çeşitli algoritmalar bulunmaktadır (Balaban Erdal M., Kartal E. 2018).

Makine öğrenmesi sınıflandırma problemlerinde, sıklıkla kullanılan algoritmalardan bazıları aşağıda verilmiştir.

2.3.1. K- en yakın komşu algoritması

K- en yakın komşu algoritması iyi bilinen ve yaygın olarak kullanılan, parametrik olmayan sınıflandırıcılardan biridir (Gou, J., Xiong, T., Kuang, Y. 2011). Genel bir ifade ile k-en yakın komşu algoritmasında yeni bir örneğin sınıfı, örneğin belirlenen bir k değerine göre, mevcut örneklem içindeki örneklere olan uzaklığı hesaplanarak (k tane en yakın komşu bulunarak) tespit edilmektedir (Balaban Erdal M., Kartal E. 2018).

Eğitimin hızlı, öğrenmenin basit ve kolay olması, gürültülü eğitim verisine gösterdiği direnç ve eğitim verisi fazla iken algoritmanın verimli olması k- en yakın komşu algoritmasının avantajlı yönleridir. “k” değerine bağlı olması, hesaplama karmaşası, hafıza kısıtı, yavaş çalışması ve ilgisiz nitelikler ile çabuk hataya düşmesi ise algoritmanın dezavantajlarındandır (Balaban Erdal M., Kartal E. 2018).

Algoritmayı kullanarak yapılan sınıflandırmanın başarısı büyük ölçüde seçilecek olan k değerine bağlıdır. Algoritmada kulllanılan k sayısı, genellikle nitelikler ve örneklem göz önünde bulundurularak geçmiş birikimlere dayanılarak seçilmektedir (Balaban Erdal M., Kartal E. 2018).

Bu algoritmada örnek sayısı n olmak üzere k, $k = \sqrt{n}$ biçiminde seçilebilmektedir (Veksler O. 2012). Birkaç k değeri ile deneme yapıldıktan sonra en az hatayı veren (en iyi performansı gösteren) sayılardan biri k değeri olarak belirlenir. (Balaban Erdal M., Kartal E. 2018).

$X = x_1, x_2, x_3, \dots, x_m$ örnek uzayı olmak üzere, bu uzaydan rastgele alınan bir $x \in X$ örneği, $< a_1(x), a_2(x), \dots, a_n(x) >$ biçiminde nitelik vektörü ile tanımlansın. Burada $a_r(x)$, x'in r. niteliğinin değerini göstermektedir (r = 1,2, ... , n) (Mitchell, T. M. 1997). Temel kNN algoritması aşağıdaki biçimde ifade edilebilir:

- Adım 1: Sınıfı belirlenmek istenen yeni örneğin, eğitim örnekleminde yer alan örneklere uzaklığı hesaplanır (Steinbach, M., Tan, P. N. 2009).
- Adım 2: Hesaplanan uzaklıklar sıralanır, içlerinden en küçük (yakın) k tanesi seçilir (Steinbach, M., Tan, P. N. 2009).
- Adım 3: Yeni örneğin sınıfının tespiti için oylama (voting) yapılır. Oylamada yaygın olan iki yaklaşım mevcuttur (Bridge, D. 2013):

Çoğunluk Oylaması

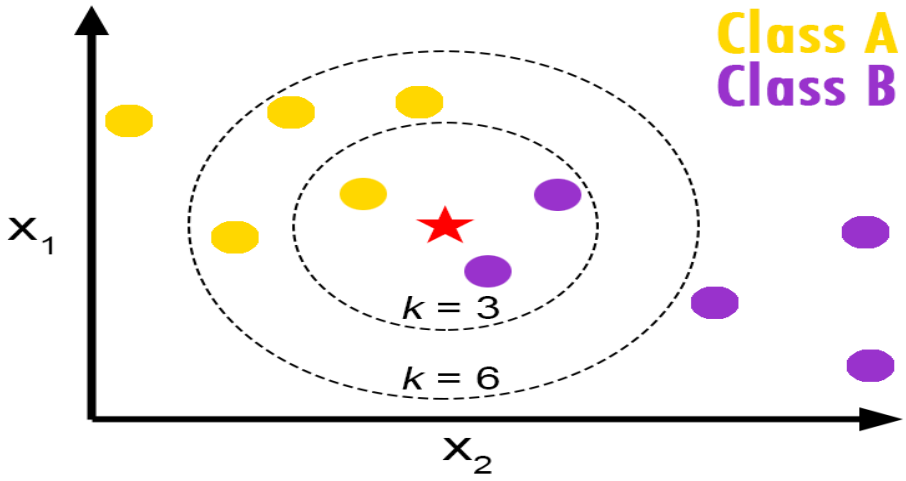
En çok tekrar eden sınıfın, yeni örneğin aranan sınıfı olduğu yöntemdir (Rudin, C. 2012).

Ters Mesafe Ağırlıklı Oylama

K- en yakın komşu algoritmasında birbirine yakın komşular daha yüksek oy alır. Ters mesafe ağırlıklı oylamasında, komşunun oyu o komşu ile sınıfı bulunmak istenen nokta (q) arasındaki uzaklığın tersi olarak belirlenir.

$$vote\ (x_i) = \begin{cases} \infty, & eğer\ d(x_i, q) = 0 \\ \dfrac{1}{d(x_i, q)}, & aksi\ taktirde \end{cases}$$

Sonunda tüm ağırlıklar toplanarak, en yüksek ağırlığa sahip sınıf seçilmektedir (Balaban Erdal M., Kartal E. 2018).



Şekil 2.8. K- en yakın komşu örneği (Venkat 2017)

K- en yakın komşu algoritmasında, tüm gözlemlerin sınıfı belirlenmek istenen q örneğine uzaklıklarının hesaplanması, bu uzaklıkların sıralanması ve k adet en küçük uzaklık değerinin bulunmasından (Şekil 2.8.) oluşan sürecin her q değeri için tekrarlanması, literatürde "Brute-Force" algoritması şeklinde adlandırılmaktadır (Verma, D., Kakkar, N., Mehan, N. 2014).

2.3.2. Naive bayes sınıflandırıcı

Adını bir matematikçi olan Thomas Bayes'ten alan "Bayes Teoremi"; olasılıkları ve gözleme dayanan olasılık dağılımlarındaki parametreleri saptama yöntemidir (McNamara, J. M., Green, R. F., Olsson, O. 2006). Bayes teoremi aşağıdaki biçimde ifade edilmektedir (Balaban Erdal M., Kartal E. 2018):

- $B_1, B_{2,}, \dots, B_k$ lar bir X örnek uzayının bir parçalanışı olsun. Burada,
- $B_i \cap B_j = \emptyset$ tüm $i \neq j$ ler için
- $\bigcup_{i=1}^{k} B_i = X$,
- $P(B_i) > 0$ tüm i ler için

ise $B_1, B_{2,}, \dots, B_k$ olaylarının X örnek uzayının parçalanışını gösterdiği unutulmamalıdır.

A, X içinde $P(A) \neq 0$ olan bir olay ise;

$P(B_r|A) = \frac{P(B_r)P(A|B_r)}{\sum_{i=1}^{k} P(B_i)P(A|B_i)}$ dir.

Naive bayes sınıflandırıcıları, verinin nasıl oluşturulduğu hakkında güçlü varsayımlar yapmakta ve bu varsayımları kapsayan bir olasılıklı model öğrenmekte; sonrasında da, meydana getirilen modelin parametrelerini tahmin etmek için, etiketlenmiş eğitim örnekleri koleksiyonunu kullanmaktadır (McCallum, A., Nigam, K. 1998).

Basit bayes sınıflandırıcı, bu modellerin içinde en basiti olup, örneklerin tüm niteliklerinin verilen sınıf bağlamında birbirinden bağımsız olduğunu varsaymaktadır. Literatürde bu varsayım "Naive Bayes Varsayımı" olarak adlandırılmıştır (McCallum, A., Nigam, K. 1998). Naive bayes, makine öğrenmesi için en verimli ve etkili tümevarımsal öğrenme algoritmalarından biridir. Bu algoritma basitliğine rağmen, çeşitli öğrenme problemlerinde iyi performans sergileyerek makine öğrenmesi araştırmacılarını şaşırtmıştır (Frank, E., Hall, M., Pfahringer, B. 2002). Naive bayes sınıflandırıcı, performans hedefi test örneklerinin sınıfını doğru bir şekilde tahmin eden ve sınıf bilgisi taşıyan eğitim örnekleri gibi danışmanlı tümevarım görevlerinde kullanılmak için tasarlanmıştır (Balaban Erdal M., Kartal E. 2018).

Naive bayes sınıflandırıcısını aşağıdaki biçimde açıklamak mümkündür (Özkan, Y. 2008):

$X = \begin{bmatrix} \overrightarrow{x_1} \\ \overrightarrow{x_2} \\ \dots \\ \overrightarrow{x_m} \end{bmatrix}$ m adet örnekten oluşan örnek uzayı, $\begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{bmatrix} \in R^{mxn}$, n adet nitelik ve m adet veriden oluşan gözlem matrisi, $c_1, c_2, \dots, c_k$ örnek uzaydaki sınıf değerleri, $\vec{x} \in X$ ise örnek uzaydan alınan ve sınıfı bilinmeyen veri örneği olmak üzere; $\vec{x}$ örneğinin sınıfını hesaplamak için;

$P(c_j|\vec{x}) = \frac{P(\vec{x}|c_j)P(c_j)}{P(\vec{x})}$ (j = 1,2, ... , k) olasılıkları hesaplanır. Örneği oluşturan nitelik değerlerinin birbirinden bağımsız olduğu kabul edilerek kullanılabilmektedir (Özkan, Y. 2008).

Sonuç olarak, sınıfı bilinmeyen bir $\overrightarrow{x_l}$ örneğinin sınıfını bulabilmek için, naiv bayes sınıflandırıcısı N gözlem sayısı ve k sınıf sayısı olmak üzere; $c_{MAP} =_{c_j} argmax \prod_{i=1}^{n} P(x_{a_i}|c_j)P(c_j)$ (i = 1,2, ..., n; j = 1,2, ..., k) olarak kullanılabilmektedir.

Bir sınıfta, bir niteliğin değeri bulunmadığında koşullu olasılığı sıfır çıkmakta, niteliğe ait diğer değerlerin olasılıkları yok sayılmaktadır. Bu nedenle, anlamlı sonuçlar elde edilememektedir. Bu sorunun üstesinden gelebilmek için ε gibi çok küçük bir değer eklenerek, hesaplamaların yapılması önerilmektedir (Özkan, Y. 2008).

Basitliği, hesaplamalardaki verimi ve iyi sınıflandırma performansı, Naive bayes sınıflandırıcının avantajları arasında yer almaktadır. Naive bayes sınıflandırıcılar, konu ile ilgili olmayan niteliklere karşı çok güçlüdür ve sınıflandırma, son tahmini yapmak için birçok nitelikten bulguyu göz önünde bulundurmaktadır (Balaban Erdal M., Kartal E. 2018).

2.3.3. Perceptron

Perceptron algoritması 1958'de Birleşik Devletler Deniz Araştırmaları Ofisi tarafından finanse edilen Frank Rosenblatt tarafından Cornell Havacılık Laboratuarı'nda icat edilmiştir (Rosenblatt, F. 1958).

Algılayıcının bir programdan ziyade bir makine olması amaçlanmış, ilk uygulaması IBM 704 için yazılım olarak geliştirilmiş daha sonra "Mark 1 algılayıcı" olarak özel üretilmiş bir donanımda kullanılmıştır. Bu makine görüntü tanıma için tasarlanmıştır. Ağırlıklar potansiyometrelerle kodlanmış ve öğrenme sırasında ağırlık güncellemeleri elektrik motorları tarafından yapılmıştır (Bishop, Christopher M. 2007).

ABD Deniz Kuvvetleri tarafından düzenlenen 1958 yılındaki basın toplantısında, Rosenblatt, AI toplumu arasında ateşli bir tartışmaya neden olan Perceptron hakkında açıklamalarda bulunmuştur.

New York Times, Rosenblatt'ın ifadelerine dayanarak, Perceptronu "Yürüyebilmesi, konuşabilmesi, görebilmesi, yazabilmesi, kendi kendini üretebilmesi ve varlığının bilincinde olaması beklenebilecek bir elektronik bilgisayarın embriyosu" olarak duyurmuştur. (Olazaran, Mikel 1996).

Her ne kadar Perceptron başlangıçta umut verici görünse de, çoklu desen sınıflarını tanımak için eğitilemedikleri hızlı bir şekilde kanıtlanmıştır. Bu, sinir ağı araştırmaları alanına, iki veya daha fazla katmana sahip bir ileri beslemeli sinir ağının, tek katmanlı algılayıcılardan daha fazla işlem gücüne sahip olduğunu göstermiştir. 1969'da Marvin Minsky ve Seymour Papert'in "Perceptrons" adlı ünlü kitabı, bu ağ sınıflarının bir XOR işlevi öğrenmesinin imkansız olduğunu göstermiştir. Bu durum sinir ağı araştırmalarının ilgisinde ve finansmanında önemli bir düşüşe neden olmuştur. Sinir ağı araştırmalarının 1980'lerde yeniden canlanması on yıllık bir süreç almıştır. 1987'de orijinal metindeki bazı hataların gösterildiği ve düzeltildiği "Perceptrons - Expanded Edition" yeniden basılmıştır. Bu çalışmayla birlikte Yapay Sinir Ağları alanındaki araştırmaların popülerliliği yeniden artmıştır. Yapay sinir ağlarının temeli olan Perceptron algoritmasının geliştirilmiş şekilleri, birçok uygulamada kullanılmaktadır (Bishop, Christopher M. 2007).

Perceptron temel kavramlar

Perceptron, ayırıcı bir hiper düzlem bularak tahminlerini yapan bir sınıflandırma algoritmasıdır. Perceptron algoritmasıyla ilgili tanımlar aşağıda verilmiştir (Genevieve B. Orr 2019):

- r : Perceptron öğrenme oranıdır. Bu oran 0-1 arasındadır.
- $y = f(z)$: Bir girdi vektörü için Perceptrondan gelen çıktıyı bildirir.
- $D = \{(x_1, d_1), \dots, (x_s, d_s)\}$: "s" örneklerinin eğitim kümesidir.
    - x_j n boyutlu girdi vektörüdür.
    - d_j Perceptron için verilen girişin çıkış değeridir.
- $x_{j,i}$: j'nci öğrenme girdi vektörünün i'nci özelliğinin değeridir.

- $x_{j,0}$ = 1.
- w_i : Ağırlık vektörünün *i*'nci değeridir. *i*'nci girdi özelliğinin değeri ile çarpılmalıdır. Çünkü $x_{j,0}$ = 1 olduğundan w_0 "b" bias sabitinin yerine kullanılır.
- $w_i\ (t)$: "t" zamanında ki *i* ağırlığını ifade eder.

Lojistik regresyon gibi diğer doğrusal sınıflandırma algoritmalarının aksine, algılayıcı algoritmasında bir öğrenme hızına gerek yoktur. Bunun nedeni, güncellemenin herhangi bir sabitle çarpılmasının yalnızca ağırlıkları yeniden ölçeklendirmesi, ancak asla tahminin işaretini değiştirmemesidir (Genevieve B. Orr 2019)

Perceptron algoritması

Perceptron algoritmasının adımları aşağıda verilmiştir (Bishop, Christopher M. 2007):

- Ağırlıklar ve eşik başlatılır. Ağırlıklar 0 veya küçük bir rastgele değerle başlatılabilir.
- Eğitim setimiz D'deki her bir örnek j için x girişi ve istenilen d çıkışı üzerinde aşağıdaki adımlar uygulanır;
    - Anlık çıktı hesabı :

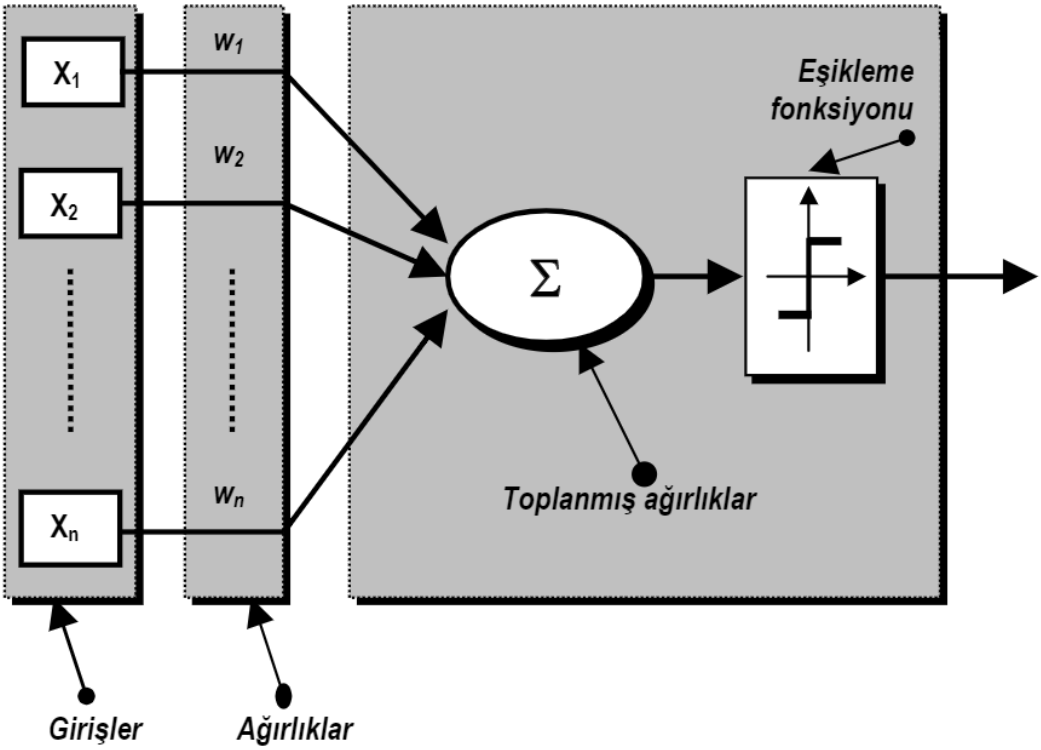
$$y_j(t) = f\left[w(t) \cdot x_j\right] = f[w_0(t)x_{j,0} + w_1(t)x_{j,1} + w_2(t)x_{j,2} + \cdots + w_n(t)x_{j,n}]$$

    - Ağırlıkların güncellenmesi :

$$w_i(t+1) = w_i(t) + r \cdot \left(d_j - y_j(t)\right) x_{j,i}$$ tüm özellikler için öğrenme oranı $0 \le i \le n, r$ olmak üzere.

- Çevrimdışı öğrenme için ikinci adım, yineleme hatasına kadar tekrarlanabilir. $\frac{1}{s}\sum_{j=1}^{s} |d_j - y_j(t)|$ kullanıcı tarafından belirlenen bir hata eşiğinden daha düşüktür. y veya önceden belirlenmiş sayıda yineleme tamamlanmıştır; burada s yine örnek setinin boyutudur.

Şekil 2.9. da bir Perceptronun yapısı gösterilmiştir.



Şekil 2.9. Bir perceptronun yapısı (Nabiyev V.V., 2016)

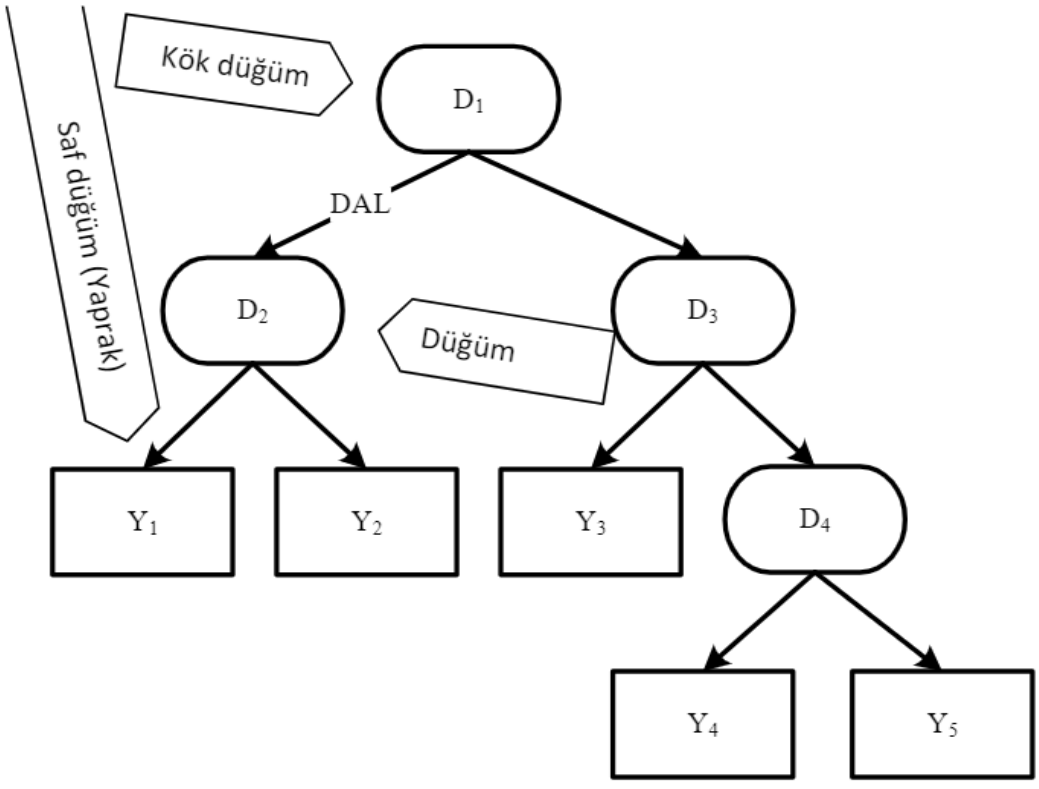
2.3.4. Karar ağaçları

Karar ağaçları yönteminde kararın verilebilmesi için ağaç biçiminde bir yapı oluşturulur (Sayad, S 2015). Karar ağaçlarında dal, yaprak gibi gerçek bir ağaçtakine benzer birtakım özel terimler kullanılmaktadır. Bir karar ağacında, karar düğümleri (decision nodes) ve yaprak düğümleri (leaf nodes) bulunur. Karar düğümleri, veri setinde karar vermek, sınıflandırma yapmak ya da tahmin etmek için kullanılan nitelikler olup, bunlar iki ya da daha fazla dala (branch) ayrılabilmektedir. Yaprak düğümleri ise kararları tutar. Ağacın en tepesinde bulunan düğüm, kök düğüm (root node) olarak adlandırılır. Bir karara ulaşabilmek için ağacın kökünden yaprak düğümlere kadar belirli bir yol (path) izlenir (Balaban Erdal M., Kartal E. 2018).

Karar ağacı oluşturulurken kullanılan genel algoritma aşağıdaki şekildedir (Şekil 2.10.) (Balaban Erdal M., Kartal E. 2018):

- Ağaç, yukarıdan aşağıya doğru yinelemeli şekilde parçala ve çöz metodu ile oluşturulur.
- İlk durumda eğitim veri setine ait tüm örnekler tek bir kökte yer alır. Bir başka ifade ile ağaç, bütün eğitim verisinin oluşturduğu tek bir düğümle başlar.
- Tahmin edici nitelikler ve hedef nitelik kategoriktir (sürekli nitelikler ayrıklaştırılarak analizlere dahil edilir).
- Eğer örneklerinin tümü aynı sınıfa aitse düğüm, yaprak olarak sonlanır ve sınıf etiketini alır.
- Eğer örneklerinin tümü aynı sınıfa ait değilse, örnekleri sınıflara en iyi bölecek nitelik seçilir.
- Nitelikler, sezgisel ya da istatiksel ölçütler temelinde seçilir.

Bir düğümdeki tüm örnekler aynı sınıfa aitse, örnekleri bölecek nitelik kalmamışsa ya da kalan niteliklerin değerini taşıyan örnek mevcut değilse karar ağacı oluşturma işlemi durdurulmaktadır (Balaban Erdal M., Kartal E. 2018).



Şekil 2.10. Örnek bir karar ağacı yapısı (Altunkaynak B. 2019)

Karar ağaçlarında ID3, C4.5, CART gibi kullanılan algoritmalar, genelleştirme hatasını en aza indirgeyerek verilen bir veri setinden bir karar ağacı çıkarmayı hedefler (Rokach, L., Maimon O. 2005). Ağaçları indüklemesi (tümevarım) için "Yukarıdan Aşağı Algoritmik Çerçeve" aşağıdaki şekilde tanımlanmıştır (Rokach, L., Maimon O. 2005):

- Her seferinde algoritma, girdi niteliklerinin ayrık bir fonksiyonunun çıktısını kullanır ve eğitim için kullanılan veri setini göz önünde bulundurur.
- En uygun fonksiyon, bazı bölme ölçülerini dikkate alarak seçilir. Bilgi kazancı, kazanç oranı ve Gini endeksi gibi tek değişkenli bölme kriterlerinin yanı sıra çok değişkenli bölme kriterleri de mevcuttur.
- Bu işlemden sonra, hiçbir bölme yeterli bir bölme ölçüsü almadıkça ya da belirlenen bir durdurma kriteri gerçekleşene kadar eğitim veri seti alt kümelere ayrılmış olur.

Karar ağaçları ile ilgili bazı avantaj ve dezavantajlar şu şekilde özetlenebilir (Zhang, K. 2006):

- Sınıflandırma ve tahmin için popüler ve güçlü araçlardır.
- İnsanlar tarafından kolay anlaşılabilen ve veri tabanı gibi bilgi sisteminde kullanılabilen kurallar yaratır ve gösterir.
- Çok fazla hesaplama olmaksızın sınıflandırma yapar.
- Sürekli ve kategorik değişkenler ile çalışma becerileri vardır.
- Tahmin ya da sınıflandırma için hangi alanların daha önemli olduğunu açıkça ortaya koyar.
- Karar ağaçları ile sınıflandırma hızlıca yapılırken, ağacın oluşturulma süresi diğer sınıflandırıcılara göre uzun zaman alabilir.
- Sürekli değerler içeren hedef değişkenlerin tahmini için uygun değildir.
- Az sayıda veri ve çok sayıda sınıf ile zayıf performans gösterir.

2.3.5. Rastgele ormanlar

Rastgele karar ormanlarının genel yöntemi ilk olarak 1995 yılında Ho tarafından önerilmiştir (Ho, Tin Kam 1995). Ho, eğik hiper düzlemlerle bölünen ağaç ormanlarının, rastgele seçilen özellik boyutlarına duyarlı olması için rastgele kısıtlandığı sürecte aşırı

zorlama olmadan büyüdüklerinde doğruluk kazanabileceğini belirlemiştir. Ho, aynı başka bir çalışmasında ise diğer bölme yöntemlerinin, bazı özellik boyutlarına karşı duyarsız olmaya zorlandığı sürece, benzer şekilde davrandığı sonucuna varmıştır (Ho TK 1998).

Bunun yanı sıra Leo Breiman'ın rastgele ormanlar kavramının erken gelişimi, tek bir ağaç yetiştirme bağlamında bir düğümü böldüğünde mevcut kararların rastgele bir alt kümesini araştırma fikrini ortaya çıkaran Amit ve Geman'ın çalışmalarından etkilenmiştir (Amit Y., Geman D. 1997).

Ho'nun rastgele alt uzay seçimi fikri, rastgele ormanların tasarımında da etkili olmuştur. Bu yöntemde bir ağaç ormanı yetiştirilir ve her bir ağaca veya her bir düğüme takılmadan önce eğitim verilerinin rastgele seçilen bir alt alana yansıtılmasıyla ağaçlar arasında değişiklik yapılır (Ho TK 1998).

Son olarak Dietterich tarafından her bir düğümdeki kararın ilk önce deterministik bir optimizasyon yerine rastgele bir prosedürle seçildiği rastgele düğüm optimizasyonu fikri ortaya atılmıştır (Dietterich, Thomas 2000).

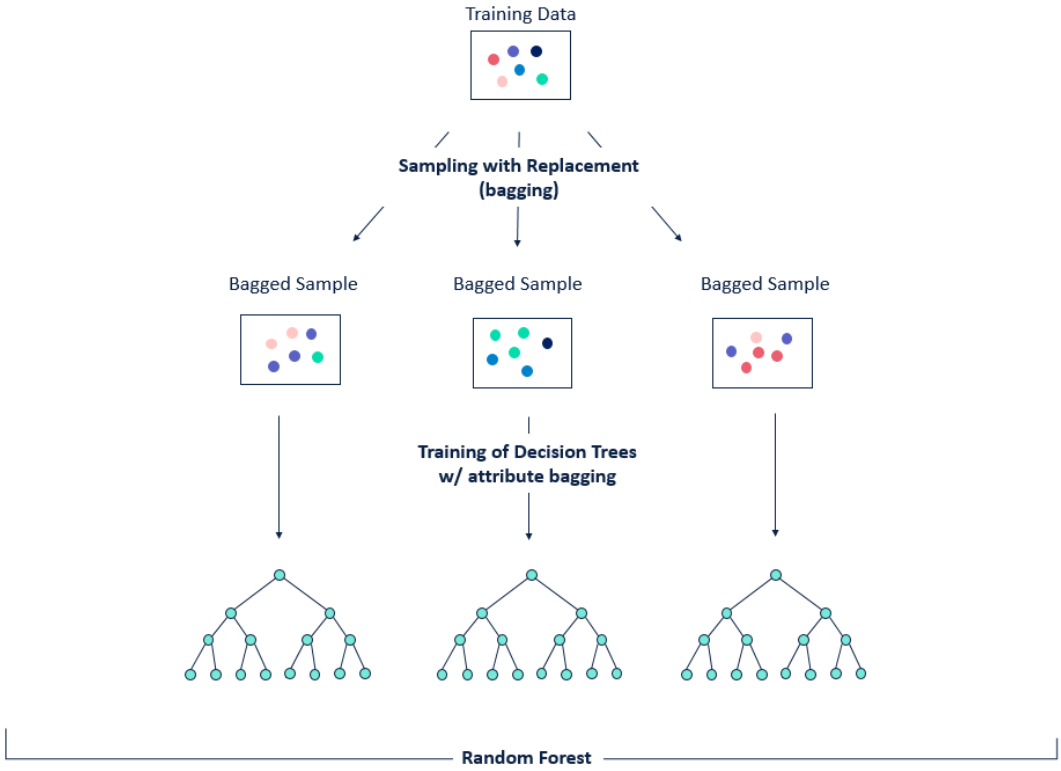
Rastgele ormanlar temel kavramlar

Rastgele ormanların tanımı ilk olarak Leo Breiman tarafından bir makalede yapılmıştır. Bu makalede, rastgele düğüm optimizasyonu ve torbalaması ile birleştirilmiş bir CART benzeri prosedür kullanılarak ilişkisiz ağaçlardan oluşan bir orman oluşturma yöntemi açıklanmıştır (Breiman L 2001). Tüm bunlara ek olarak, bu makale, özellikle rastgele ormanların modern uygulamasının temelini oluşturan, bazıları daha önce bilinen ve bazıları yeni olan çeşitli kavramları birleştirmektedir.

Bu temel kavramlar :

- Genelleme hatasının (generalization error) tahmini olarak torbadan çıkma (out-of-bag) hatası kullanılması,
- Permütasyon yoluyla değişken önemi ölçme.

Ayrıca bu çalışma ormandaki ağaçların gücüne ve korelasyonlarına bağlı olarak, genelleme hatasına bağlılık şeklinde rastgele ormanlar için ilk teorik sonucu da sunmaktadır. Şekil 2.11. de rastgele orman örneği gösterilmiştir.



Şekil 2.11. Rastgele orman örneği (Sydney, F. 2019)

Rastgele orman algoritması

Genel olarak algoritma karar ağaçlarının eğitimi, bagging (torbalama) işlemi ve rastgele alt uzay metodu olmak üzere 3 adımdan oluşmaktadır. Rastgeleleştirmeye bir adım daha eklenmesi aşırı derecede rastgele ağaçlar veya ekstra ağaçlar verir. Ekstra ağaçlar (extra trees) adımı 3 adımın ardından ayrıca açıklanacaktır.

Karar ağaçlarının eğitimi

Karar ağaçları, çeşitli makine öğrenimi uygulamaları için popüler bir yöntemdir. Karar ağaçları anlaşılması ve kullanılması açısından oldukça kolaydır. Özellikle, çok derin

büyüyen ağaçlar, oldukça düzensiz desenleri öğrenme eğilimindedir. Rastgele ormanlar yöntemi, varyansı azaltmak amacıyla aynı eğitim setinin farklı bölümlerinde eğitilmiş birden fazla derin karar ağacının ortalamasını alma yoludur (Hastie Trevor, Tibshirani Rober, Friedman Jerome 2008). Bu yorumlanabilirlik açısından bir kayba neden olsa da genellikle son modeldeki performansı, büyük ölçüde arttırmaktadır.

Torbalama (bagging) metodu

Rastgele ormanlar için genel eğitim algoritması literatürde torbalama metodu olarak adlandırılmıştır. $X = x_1, \dots, x_n$ bir eğitim seti $Y = y_1, \dots, y_n$ cevaplar olmak üzere tekrar tekrar torbalama (B kez) eğitim setinin değiştirilmesi ile rastgele bir örnek seçilir. Seçilen örneklere uygun ağaçlar oluşturulur. $b = 1, \dots, B$ olmak üzere (Hastie Trevor, Tibshirani Rober, Friedman Jerome 2008):

- X,Y den n eğitim örneği alınarak tekrar edilir. Bu örnekler X_b, Y_b olsun,
- f_b sınıflandırma eğitimi X_b, Y_b de uygulanır.

Eğitimden sonra, görülmeyen örnekler x' için tahminler x' üzerindeki tüm bireysel regresyon ağaçlarından tahminlerin ortalaması alınarak yapılır.

$$\hat{f} = \frac{1}{B} \sum_{b=1}^{B} f_b\,(x')$$

Bu yöntem ile daha iyi bir performans elde edilir. Bunun nedeni, sapmayı arttırmadan modelin varyansının azalmasıdır. Tek bir ağacın tahminleri eğitim setindeki gürültüye karşı oldukça hassas olmasına rağmen, ağaçlar korelasyon halinde olmadığından bu hassasiyet birçok ağacın ortalaması kadar değildir.

Rastgele alt uzay (random subspace) metodu

Rastgele ormanlar, öğrenme sürecindeki her adayda özelliklerin rastgele bir alt kümesini seçen, değiştirilmiş bir ağaç öğrenme algoritması kullanır. Bu işleme "özellik torbalama"

(feature bagging) denir. Bunun nedeni, sıradan bir önyükleme örneğindeki ağaçların korelasyonu olmasıdır.

Tipik olarak, p unsurlarıyla ilgili bir sınıflandırma problemi için, her bölmede √p (yuvarlatılmış) sayıda özellik kullanılmaktadır. Regresyon problemleri için ise varsayılan olarak minimum düğüm büyüklüğü 5 olan p/3 (yuvarlatılmış) sayıda özellik kullanılması önerilmektedir. Uygulamaya göre seçilecek olan özellikler değişkenlik gösterir (Hastie Trevor, Tibshirani Rober, Friedman Jerome 2008).

Ekstra ağaçlar (extra trees)

Sıradan rastgele ormanlara benzer olarak, tek tek ağaçların bir topluluğu oldukları için, iki ana fark vardır: Birincisi, her ağaç tüm öğrenme örneği (bir bootstrap örneği yerine) kullanılarak eğitilir. İkinci fark ise ağaç öğreniminde yukarıdan aşağıya bölünme rastgeleleştirilmesidir. Göz önünde bulundurulan her özellik için yerel olarak optimal kesme noktasını hesaplamak yerine rastgele bir kesme noktası seçilir. Bu değer, özelliğin ampirik aralığı içindeki (ağacın eğitim setinde) eşit dağılımından belirlenir. Daha sonra, rastgele oluşturulan tüm bölünmelerden en yüksek puanlı bölünme düğümü, bölme için seçilir. Sıradan rastgele ormanlara benzer şekilde, her bir düğümde dikkate alınacak rastgele seçilen özelliklerin sayısı belirlenebilir. Bu parametre için varsayılan değerler $\sqrt{n}$, regresyon problemleri için ise n'dir; burada n, modeldeki özellik sayısıdır (Geurts P, Ernst D, Wehenkel L 2006).

2.3.6. Gradient boosting yöntemi

Gradient boosting fikri, Leo Breiman'ın artırmanın uygun bir maliyet fonksiyonu üzerinde bir optimizasyon algoritması olarak yorumlanabileceği gözleminden doğmuştur (Breiman, L. 1997). Daha sonra Jerome H. Friedman tarafından açık regresyon gradient boosting algoritmaları geliştirilmiştir (Friedman, J. H. 1999). Bunlara eş zamanlı olarak Llew Mason, Jonathan Baxter, Peter Bartlett ve Marcus Frean tarafından daha genel bir fonksiyonel gradient boosting kavramı ortaya atılmıştır (Mason, L., Baxter, J., Bartlett, P. L., Frean, Marcus 1999). Bu çalışmalarda yinelemeli fonksiyonel gradyan iniş

algoritmaları olarak yükseltme algoritmalarının görünümü anlatılmıştır. Bu algoritmalar negatif gradyan yönünü gösteren bir işlevi yinelemeli olarak seçerek, maliyet fonksiyonunu işlev alanı üzerinde optimize eden algoritmalardır. Gradient boosting metodu, makine öğreniminin birçok alanında, regresyon ve sınıflandırma problemlerinde kullanılmaktadır.

Gradient boosting temel kavramlar

Diğer boosting yöntemleri gibi, gradient boosting yöntemi de zayıf "öğrenenleri" tek bir güçlü öğrenicide yinelemeli bir şekilde birleştirmektedir. Bunu açıklamanın en kolay yolu küçük karaler regresyon yöntemidir. Burada amaç ortalama kare hatasını en aza indirerek $\frac{1}{n}\sum_i(\hat{y}_i - y_i)^2$, $\hat{y} = F(x)$ modelinin değerlerini tahmin etmeyi bir F modeline öğretmektir (Cheng Li 2019).

Her aşamada gradient boosting için m, $1 \leq m \leq M$ olmak üzere F_m'in kusurlu bir model olduğu varsayılmaktadır. Çünkü başlangıçta, eğitim setindeki ortalama y'yi tahmin eden çok zayıf bir model kullanılabilir. Gradient boosting algoritması daha iyi bir model oluşturmak için bir h tahminleyicisi ekleyerek F_m üzerinde yeni bir model oluşturur (Cheng Li 2019). Bu model $F_{m+1}(x) = F_m(x) + h(x) = y$ şeklindedir. (Cheng Li 2019).

Gradient boosting algoritması

Danışmanlı birçok öğrenme probleminde, bir çıkış değişkeni y ve bir ortak olasılık dağılımı P (x, y) ile tarif edilen, giriş değişkenleri olan x'in bir vektörü vardır. $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, bilinen x değerlerine karşılık gelen y değerleri olmak üzere verilen eğitim setinde amaçlanan, belirli bir kayıp fonksiyonunun L(y,F(x)) beklenen değerini en aza indiren F(x) fonksiyonuna ait bir $\hat{F}(x)$ bulmaktır (Cheng Li 2019).

Giriş eğitim seti $\{(x_{i,}y_{i,})\}_{i=1}^{n}$, farklılaşabilir kayıp fonksiyonu L(y,F(x)), iterasyon sayısı M olmak üzere;

- Model sabit bir değerle başlatılır.

$$F_0(x) = arg_\gamma min \sum_{i=1}^{n} L(y_i, \gamma)$$

- m = 1, M pseudo-residuals hesaplanır.

$$r_{im} = -[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}]_{F(x)-F_{m-1}(x)} \qquad i = 1, \dots, n \; için$$

$$\gamma_m = arg_\gamma min \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$$

- Model güncellenir.

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

- Çıkış $F_M(x)$ olur.

Gradient tree boosting metodu

Gradient boosting yöntemi tipik temel öğrenen olarak, sabit boyutlu karar ağaçları (özellikle CART ağaçları) ile kullanılmaktadır. Bu özel durum için Friedman, her temel öğrenenin uyum kalitesini artıran gradient boosting yönteminde bir değişiklik önermiştir. Friedman, tüm ağaçlar için tek bir γ_m yerine, ağaç bölgelerinin her biri için ayrı bir optimum değer γ_{jm} seçmeyi önermiş ve değiştirilmiş bu algoritmayı "tree boosting" olarak adllandırmıştır (Friedman, J. H. 1999). Gradient tree boosting algoritması bu değişiklikle aşağıdaki şekle dönüşmüştür:

$F_m(x) = F_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} 1R_{jm}(x)$,
$\gamma_{jm} = arg_\gamma min \sum_{x_i \epsilon R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma)$.

3. MATERYAL VE YÖNTEM

3.1. Endüstriyel Otomasyon Sistemlerinde Arıza Sınıflandırması

Endüstride kullanılan otomasyon makinelerinde üretim süreçleriyle birlikte zamanla çeşitli arıza durumları meydana gelmektedir. Fabrikalarda makinelerde oluşan arızalar bakım ekiplerinin müdahalesi sonrası tespit edilerek çözülmektedir. Bakım ekiplerinin arıza durumlarının gerçekleşmesi sonucu uyguladığı ilk adım, arızanın tespit edilmesidir. Tespit edilen arıza girderildikten sonra bakım personeli arızaya yönelik kök-neden analizini yaparak arızaya neden olan ana sebebi bulur. Bu sebeple birlikte arıza ve sonucunu raporlar.

Makine üzerinde meydana gelen arızalar genel anlamda mekanik, elektrik, pnömatik, hidrolik vb. ana kategoriler altında sınıflandırılmaktadır. Bu ana kategoriler alt kırılımlara ayrılarak her bir ana arıza nedeni çeşitli alt arıza dallarına ayrılmaktadır. Örnek olarak bir makine üzerinde yaşanan bir pnömatik silindirin hareket etmemesinin nedeni piston üzerindeki bir sensör arızası ise bu elektrik ana kategorisi altındaki sensör alt dal arızasına karşılık gelmektedir. Eğer gerekli görülmüşse bir fabrikada kullanılan sensör tipleride kategorize edilerek sensör arızaları da kendi içinde alt kırılımlara ayrılabilmektedir.

Makinelerde meydana gelen arızalar analiz edilerek, sorun yaşanan önemli noktalar tespit edilip kalıcı çözümler uygulayarak arıza tekrarlarının önüne geçmek, bakım ekiplerinin asli görevlerindendir. Bu süreçlerde yaşanan arızalara yönelik incelemeler, bakım personellerinin tespitlerinden oluşan verilerden analizler yapılarak gerçekleştirilmektedir. Zamanla bir makine üzerinde oluşan arızaların kodlanması ve oluşan deneyimin yaşanan arıza çeşitleriyle birlikte artması, makine üzerindeki arıza tespiti ve çözümlerinin hızlı bir şekilde gerçekleşmesi açısından önemlidir.

Zaman zaman makine üzerinde birden fazla arıza aynı anda oluşabilmektedir. Bu tip durumlarda bakım personelinin bilgi birikimi ile birlikte tespitleri sonucunda arızalar sırayla tespit edilerek giderilmektedir. Ancak yaşanan birden fazla arızanın tespiti ve çözümü her zaman aynı sıra ve zamanda gerçekleştirilememektedir. Burada arızaya

müdahale süreleri, bakım ekiplerine verilen eğitim ile hemen hemen aynı seviyelere çekilebilse de arızanın tespiti noktasında kişilerin deneyimleri ön plana çıkmaktadır. Bu da arıza tespiti noktasında yaşanan zaman kayıpları ya da kök neden tespitlerindeki yanlışlıklara yol açmaktadır. Otomasyon sistemlerinde, gelişen SCADA sistemleri ile birlikte makine üzerindeki yaşanabilecek çeşitli arıza durumları kodlanarak otomatik olarak makine ekranlarında gösterilebilmektedir. Ancak yine de yaşanan çoğu arızanın tespitinde, özellikle yüksek giriş çıkış sayısına sahip, gelişmiş mekanik, hidrolik ve pnömatik sistemlere sahip makinelerde arıza tespitleri bakım ekibinin analizleri sonucunda gerçekleştirilebilmektedir.

Tüm bu konular çerçevesinde yaşanılan arıza durumlarının tespiti bir makine öğrenmesi sınıflandırma problemine benzemektedir. Yaşanılan arıza durumunda makine üzerinde yer alan sistemlerin, arızanın yaşanmasına sebep olan kısımlarının, makinenin normal çalışma düzeninden farklı bir durumda olduğu düşünülebilir. Bu durumda makine üzerinden gerekli ve yeterli sayıda sinyal alınabilmesi durumunda, yaşanan arıza durumunda belirli sayıda bir veri elde etmek mümkündür.

Yapısı itibariyle elde edilen verilerden oluşturulan bir veri setinin danışmanlı eğitime uygun olduğu görülmektedir. Çünkü oluşan her arıza durumundaki elde edilen verilerin, danışmanlı eğitimdeki etiket (label) olarak nitelendirdiğimiz sonuca karşılık gelen bir arıza kodu veya tanımı vardır. Bunu bakım personelinin o arıza durumundaki makinenin durumunu öğrenmesine örnek verebiliriz. Bu da uygun bir sınıflandırma algoritmasıyla birlikte makine üzerindeki arızanın otomatik tespitinin yapılabileceğini göstermektedir.

Arıza tespitiyle birlikte arıza tahminlerinin yapılması da önemli bir durumdur. Arıza oluşmadan önce arızanın tahmininin yapılabilmesi sayesinde alınacak önlemler ile uzun süreli makine duruşlarının önüne geçilmesi de sağlanmış olacaktır. Bakım ekipleri tarafından kestirimci bakım adı altında çeşitli parçaların ömür takipliliği, bazı titreşim akım verilerini ölçerek bozulmaya yaklaşan makine aksamlarının onarımları yapılmaktadır.

Makine üzerinden alınabilecek doğru verilerle, oluşabilecek arızaların önceden tahmin edilmesini sağlayacak yapay zeka modeli geliştirilebilir. Günümüzde arıza tespiti ve tahminine yönelik birçok araştırma ve proje çalışmaları bulunmaktadır.

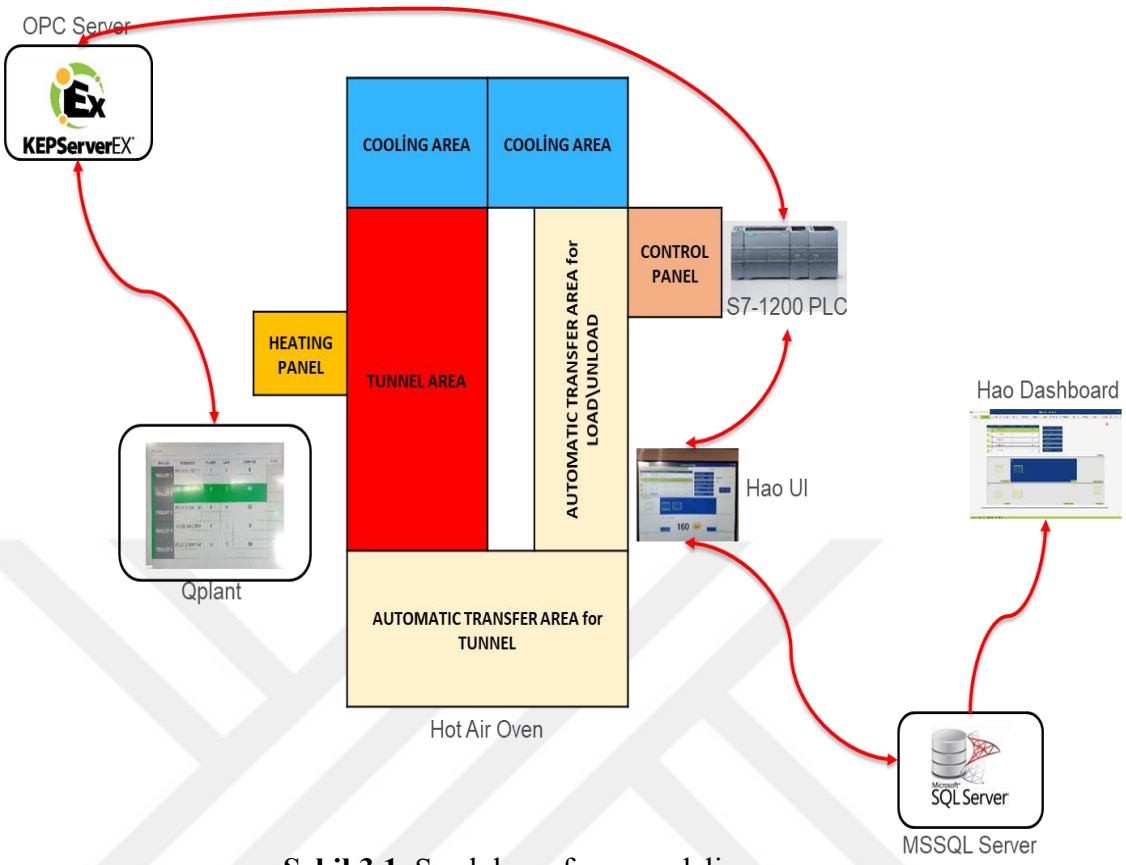
Titreşim sinyallerinden alınan verilerle makineler üzerindeki motorlarda oluşabilecek arızaların önceden tahmin edilmesi, seri üretim hatları üzerinden alınan verilerle üretim hattının durmasına neden olan en sık rastlanan arızanın önceden tespiti, robotlar üzerinden alınan sinyaller ile robotun arıza durumuna geçmesine neden olabilecek hatanın önceden tespit edilmesine yönelik çeşitli uygulamalar ve araştırma tezleri yapay zeka teknikleri kullanılarak gerçekleştirilmiştir.

Son olarak İstanbul Teknik Üniversitesinin Trakya Elektrik Dağıtım şirketi ile başlayacağı yeni bir projede öngörülü bakım teknolojisini geliştirmek hedeflenmektedir (Anonim, 2019e). Proje kapsamı big data ve yapay zekadan yararlanarak enerji hatlarında oluşabilecek arızanın önceden tespit edilmesi ve bakım önlemlerinin alınması olarak belirlenmiştir. Böylece kesintisiz bir elektrik dağıtımı ve enerji verimliliği sağlanmış olacaktır.

Bu tez çalışması kapsamında, yapay zeka yöntemleriyle endüstriyel otomasyon sistemlerinde oluşan arızaların tespit ve tanımlamasının yapılması hedeflenmektedir. Bu sayede bakım ekibinin arızaya olan müdahelesinin hızlandırılması ve arızanın kök nedeninin sağlıklı bir şekilde tespit edilmesi sağlanacaktır. Böylelikle makine duruş süreleri kısalacak ve arızaya yönelik daha kalıcı çözümler üretilebilecektir.

3.2. Örnek Endüstriyel Otomasyon Makinesi Genel Bilgiler

Bu tez çalışmasında yapay zeka yöntemlerinin uygulanacağı örnek endüstriyel otomasyon sistemi olarak şu an bir fabrikada kullanılmakta olan sıcak hava fırını seçilmiştir. Sıcak hava fırını, yakıt borularına form vermek amacıyla işlem yapmaktadır. Seçilen sıcak hava fırınında Tünel Bölgesi, Duşlama Bölgesi ve Transfer Bölgesi olmak üzere 3 temel kısım bulunmaktadır.



Şekil 3.1. Sıcak hava fırını modeli

Şekil 3.1. de sıcak hava fırınının genel modeli görülmektedir. Transfer bölgesindeki sarım alanında kalıba sarılan yakıt boruları tünel bölgesinde ısıl işlem görerek (prosese bağlı olarak 180 °C - 220 °C arası sıcaklıkta) daha sonra duşlama bölgesinde 10 °C ‘ lik suyla şoklama işlemi görüp transfer bölgesine form almış şekilde ulaşmaktadır. Kalıbın tünel dışında taşınma işlemleri pnömatik yolla çalışan pistonlar sayesinde sağlanmaktadır.

Tünel bölgesinde konveyör amaçlı bir zincir bulunmaktadır. Ayrıca alt ve üst rezistans gruplarına uygulanan PID işlemi ile tünel set edilen optimum sıcaklıkta ısısını dengelemektedir. PT100 aracılığıyla fırının 3 noktasından sıcaklık ölçümü yapılmaktadır. Zincirin haraketini sağlayan bir sürücü, motor ve redüktör bulunmaktadır.

Duşlama bölgesinde kalıbı tünelden çıkartan ve daha sonra duşlama kısmına götüren pistonlar bulunmaktadır. Duşlama kısmında su fıskiyeleri ve duşlanan suyun toplandığı su tankı bulunmaktadır.

Transfer bölgesinde kalıbın hareketini sağlayan pistonlar, pistonların üzerinde hareketi sağlayan mekanik taşıma dilleri, ileri –geri konumların tespitini sağlayan manyetik sensörler bulunmaktadır. Ayrıca tünel dışında kalıpların durduğu çeşitli istasyonlar mevcuttur. Bu istasyonlara ait kalıbın varlığını kontrol eden indüktif sensörler bulunmaktadır.

Sıcak hava fırını otomasyonel anlamda PID ile ısıl işlem dengesinin kontrol edildiği, piston haraketlerinin otomatize edilmesiyle kalıpların taşındığı, duşlama kontrolüyle suyun kontrol edildiği bir temel otomasyonda çalışmaktadır.

Bu temel otomasyon çalışmalarının yanında sıcak hava fırınında uygulanan dijital dönüşüm projesi ile birlikte fırına ait bir Scada yazılımı yapılmıştır. Fırın 15 sanal bölgeye ayrılarak bu ekran üzerinden kalıpların hangi bölgede olduğunun takibi yapılmaktadır. Anlık sıcaklıklar, tank seviyesi ve duşlama suyu sıcaklıkları bu ekrandan görülebilmektedir. Bunun yanı sıra fırının tüm manuel hareketleri bu ekran üzerinden yapılabilmektedir. Aynı zamanda fırın PLC ve Scada ekranlarına yapılan özel yazılım geliştirmesi ile fırına ait girilecek parametreler kalıp üzerindeki etiketin sisteme okutulması sonucu otomatik olarak sunucu üzerinden fırına yüklenmektedir. Proses paramatereleri fırının sunucu ile haberleşmesi ile scada sistemi tarafından otomatik yüklenmektedir. Ayrıca her kalıp sarım bölgesinde etikek okutulma yolu ile kalitesel olarak parametre kontrolüne tabi tutulmaktadır. Bunun yanı sıra kalıpların dengeli bir ısıl işlem görmesi amacı ile scada sisteminden PLC sistemine aktarılan kalıp boyları hesaplama algoritmasıyla tünel bölgesine girecek kalıpların uygun zaman ile tünele alınması otomatik olarak sağlanmaktadır.

Fırında uygulanan dijital dönüşüm ile sunucuda fırına ait üretim kayıtları, duruş kayıtları ve proses kriter bilgileri anlık olarak kaydedilmektedir. Endüstri 4.0 süreçlerine uygun bir gelişim ile operatöre minimum seviyede bağlılık sağlanan bu makine uzaktan bir uygulama aracılığı ile de anlık durumu takip edilebilmektedir.

Bu tez çalışmasında bu makine modeli baz alınarak makine üzerinde gerçekleşen arıza kayıtları tespit edilmiş ve bu arızaya sebep olan senaryolardaki ölçülmesi ve kontrol

edilmesi gereken noktalar tespit edilerek veri seti oluşturmak amacıyla makine modeli bir uygulamaya aktarılmıştır. Bu uygulamadan Data Set Form Uygulaması başlığı altında detaylı olarak bahsedilecektir.

Makine üzerinde gerçekleşebilen arıza tipleri

Sıcak hava fırını üzerinde meydana gelebilen arızalar incelenerek fırının duşlama, tünel ve transfer bölgeleri olarak gruplanmış, oluşabilecek arıza tanımları ve bunlar için oluşturulan arıza kodları aşağıda gösterilmiştir (Çizelge 3.1., Çizelge 3.2., ve Çizelge 3.3.). Bu arıza kodları veri setimizin çıktıları olarak düşünülebilir. Uygulayacağımız öğrenme metotlarının çıktıları bu arıza kodları arasından olacaktır.

Çizelge 3.1. Transfer bölgesi arıza tanımları ve kodları

Arıza Tanımları	Arıza Kodları
4.Piston arızası	18
4.Piston ileri konum sensör arızası	19
4.Piston geri konum sensör arızası	20
4.Piston valf arızası	21
4.Piston hava kaçağı	22
4.Piston taşıma dili arızası	23
5.Piston arızası	24
5.Piston ileri konum sensör arızası	25
5.Piston geri konum sensör arızası	26
5.Piston valf arızası	27
5.Piston hava kaçağı	28
5.Piston taşıma dili arızası	29
6.Piston arızası	30
6.Piston ileri konum sensör arızası	31

Çizelge 3.1. Transfer bölgesi arıza tanımları ve kodları (Devam)

Arıza Tanımları	Arıza Kodları
6.Piston geri konum sensör arızası	32
6.Piston valf arızası	33
6.Piston hava kaçağı	34
6.Piston taşıma dili arızası	35
7.Piston arızası	36
7.Piston ileri konum sensör arızası	37
7.Piston geri konum sensör arızası	38
7.Piston valf arızası	39
7.Piston hava kaçağı	40
7.Piston taşıma dili arızası	41
Kilitleme pistonu arızası	42
Kilitleme pistonu ileri konum sensör arızası	43
Kilitleme pistonu geri konum sensör arızası	44
Kilitleme piston valf arızası	45
Kilitleme piston hava kaçağı	46
Kilitleme mekanizması arızası	47
4.İstasyon sensörü arızası	51
5.İstasyon sensörü arızası	52
6.İstasyon sensörü arızası	53
7.İstasyon sensörü arızası	54
Tünel giriş sensör arızası	55

Çizelge 3.2. Duşlama bölgesi arıza tanımları ve kodları

Arıza Tanımları	Arıza Kodları
1.Piston arızası	0
1.Piston ileri konum sensör arızası	1
1.Piston geri konum sensör arızası	2
1.Piston valf arızası	3
1.Piston hava kaçağı	4
1.Piston taşıma dili arızası	5
2.Piston arızası	6
2.Piston ileri konum sensör arızası	7
2.Piston geri konum sensör arızası	8
2.Piston valf arızası	9
2.Piston hava kaçağı	10
2.Piston taşıma dili arızası	11
3.Piston arızası	12
3.Piston ileri konum sensör arızası	13
3.Piston geri konum sensör arızası	14
3.Piston valf arızası	15
3.Piston hava kaçağı	16
3.Piston taşıma dili arızası	17
Tünel çıkış sensör arızası	48
2.İstasyon sensörü arızası	49
3.İstasyon sensörü arızası	50
Fıskiye arızası	56
Pompa arızası	57

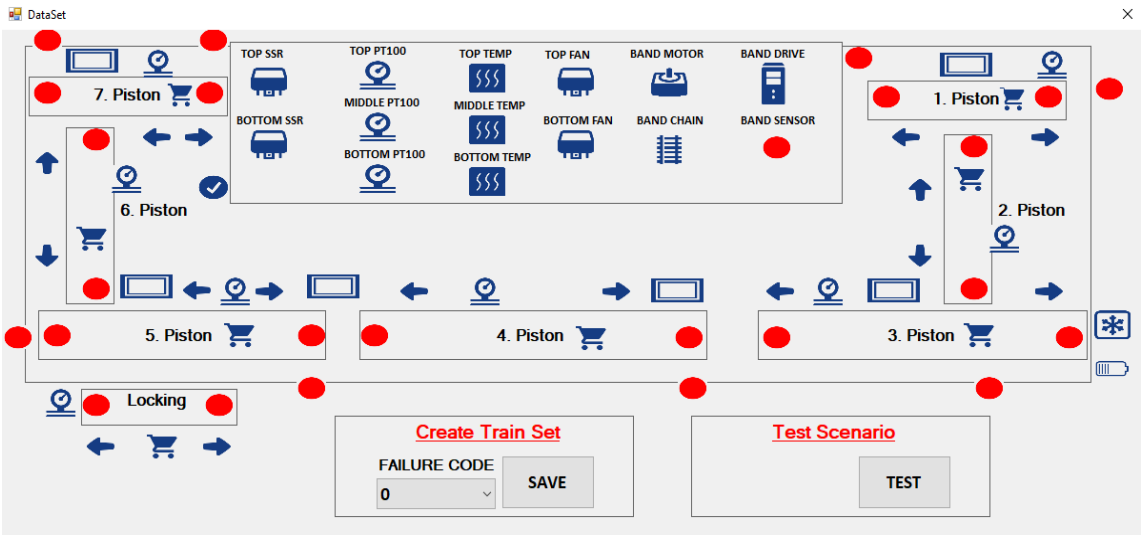
Çizelge 3.3. Tünel bölgesi arıza tanımları ve kodları

Arıza Tanımları	Arıza Kodları
Üst rezistans grubu arızası	58
Alt rezistans grubu arızası	59
Üst PT100 arızası	60
Alt PT100 arızası	61
Orta PT100 arızası	62
Alt fan arızası	63
Üst fan arızası	64
Bant motoru arızası	65
Bant motor sürücüsü arızası	66
Bant zincir atması arızası	67
Konveyör dişli ve rulman arızası	68
Üst solid state röle arızası	69
Alt solid state röle arızası	70

3.3. Veri Seti Form Uygulaması

3.3.1. Form uygulaması genel bilgiler

Bu tez çalışmasında oluşabilecek arızalara ait bir veri seti oluşturabilmek amacıyla DataSet adında bir uygulama geliştirilmiştir. Bu uygulamaya ait ekran görüntüsü aşağıda verilmiştir.



Şekil 3.2. Veri seti uygulaması

Bu uygulama aracılığıyla, arıza durumunda oluşması muhtemel çeşitli senaryolar için fırında ölçüm alınabilen noktalardan gelecek olan sinyallerin durumları sembollize edilebilmektedir. Bu semboller üzerinden senaryo kodlanarak veri setinin oluşturulması sağlanmıştır. Semboller üzerinden kodlanan arıza durumu, arıza kodu seçilerek bu uygulama aracılığı ile veri tabanına kaydedilmiştir. Her kayıt öncesi mevcut kodlanan durumun veri tabanında daha önce kaydedilip kaydedilmediği kontrol algoritmasıyla kontrol edilip aynı senaryo bulunursa hata uyarısı verilmektedir. Böylece eğitim için sağlıklı bir veri seti oluşturulması hedeflenmiştir.

Ayrıca eğitim süreci tamamlandıktan sonra aynı uygulama kullanılarak eğitim veri setinde kullanılmamış olan bir arıza senaryosu oluşturulup test butonuna basıldığında öğrenme algoritması çıktısı ekran üzerinden görülebilmektedir. Burada test için oluşturulan senaryonun eğitim setinde olup olmadığı da kontrol edilerek eğer aynı kayıt veri tabanında varsa kullanıcıya uyarı verilmektedir. Böylece sağlıklı bir test işlemi yapılabilmektedir.

3.3.2. Form uygulaması makine modellemesi

Veri seti form uygulaması üzerinde sıcak hava fırınında gerçekleşebilecek arıza durumunlarında değerlendirmeye alınacak 79 adet sinyal, semboller aracılığıyla modellenmiştir. Modellenen ölçüm sinyal noktaları, kodları ve sembolleri aşağıda ki çizelgelerde gösterilmiştir (Çizelge 3.4., Çizelge 3.5., Çizelge 3.6.).

Çizelge 3.4. Transfer bölgesi kontrol noktaları, kodları ve sembolleri

Kontrol noktaları tanımları	Kontrol noktaları kodları	Kontrol noktaları sembolleri (Durum 0 ise)	Kontrol noktaları sembolleri (Durum 1 ise)
3.Bölge kalıp varlık durumu	17		
4.Piston ileri konum sensörü	29		
4.Piston geri konum sensörü	30		
4.Piston taşıma dili durumu	31		
4.Piston hava kaçağı durumu	32		
4.Piston ileri hareket valf çıkışı	33		
4.Piston geri hareket valf çıkışı	34		
4.İstasyon sensörü	18		
4.Bölge kalıp varlık durumu	27		
5.Piston ileri konum sensörü	37		

Çizelge 3.4. Transfer bölgesi kontrol noktaları, kodları ve sembolleri (Devam)

Kontrol noktaları tanımları	Kontrol noktaları kodları	Kontrol noktaları sembolleri	Kontrol noktaları sembolleri
5.Piston geri konum sensörü	38		
5.Piston taşıma dili durumu	39		
5.Piston hava kaçağı durumu	40		
5.Piston ileri hareket valf çıkışı	41		
5.Piston geri hareket valf çıkışı	42		
5.İstasyon sensörü	28		
5.Bölge kalıp varlık durumu	35		
6.Piston ileri konum sensörü	45		
6.Piston geri konum sensörü	46		
6.Piston taşıma dili durumu	47		
6.Piston hava kaçağı durumu	48		
6.Piston ileri hareket valf çıkışı	49		
6.Piston geri	50		
6.İstasyon sensörü	36		
6.Bölge kalıp varlık durumu	43		
7.Piston sensörü	53		

Çizelge 3.4. Transfer bölgesi kontrol noktaları, kodları ve sembolleri (Devam)

Kontrol noktaları tanımları	Kontrol noktaları kodları	Kontrol noktaları sembolleri (Durum 0 ise)	Kontrol noktaları sembolleri (Durum 1 ise)
7.Piston geri konum sensörü	54		
7.Piston taşıma dili durumu	55		
7.Piston hava kaçağı durumu	56		
7.Piston ileri hareket valf çıkışı	57		
7.Piston geri hareket valf çıkışı	58		
7.İstasyon sensörü	44		
Tünel giriş sensörü	51		
Kilitleme pistonu ileri konum sensörü	59		
Kilitleme pistonu geri konum sensörü	60		
Kilitleme pistonu mekanizması durumu	61		
Kilitleme pistonu hava kaçağı durumu	64		
Kilitleme pistonu ileri hareket valf çıkışı	62		
Kilitleme pistonu geri hareket valf çıkışı	63		
Tünele kalıp giriş onay durumu	52		

Çizelge 3.5. Duşlama bölgesi kontrol noktaları, kodları ve sembolleri

Kontrol noktaları tanımları	Kontrol noktaları kodları	Kontrol noktaları sembolleri (Durum 0 ise)	Kontrol noktaları sembolleri (Durum 1 ise)
Tünel çıkış sensörü	0		
1.Bölge kalıp varlık durumu	1		
1.Piston ileri konum sensörü	3		
1.Piston geri konum sensörü	4		
1.Piston taşıma dili durumu	5		
1.Piston hava kaçağı durumu	6		
1.Piston ileri hareket valf çıkışı	7		
1.Piston geri hareket valf çıkışı	8		
2.İstasyon sensörü	2		
2.Piston ileri konum sensörü	11		
2.Piston geri konum sensörü	12		
2.Piston taşıma dili durumu	13		
2.Piston hava kaçağı durumu	14		
2.Piston ileri hareket valf çıkışı	15		
2.Piston geri hareket valf çıkışı	16		

Çizelge 3.5. Duşlama bölgesi kontrol noktaları, kodları ve sembolleri (Devam)

Kontrol noktaları tanımları	Kontrol noktaları kodları	Kontrol noktaları sembolleri (Durum 0 ise)	Kontrol noktaları sembolleri (Durum 1 ise)
3.İstasyon sensörü	10		
2.Bölge kalıp varlık durumu	9		
3.Piston ileri konum sensörü	19		
3.Piston geri konum sensörü	20		
3.Piston taşıma dili durumu	21		
3.Piston hava kaçağı durumu	22		
3.Piston ileri hareket valf çıkışı	23		
3.Piston geri hareket valf çıkışı	24		
Duşlama akış sensörü	25		
Tank seviyesi sensörü	26		

Çizelge 3.6. Tünel bölgesi kontrol noktaları, kodları ve sembolleri

Kontrol noktaları tanımları	Kontrol noktaları kodları	Kontrol noktaları sembolleri (Durum 0 ise)	Kontrol noktaları sembolleri (Durum 1 ise)
Üst solid state röle durumu	65		
Alt solid state röle durumu	66		
Üst konum PT100 sensörü	67		
Orta konum PT100 sensörü	68		
Alt konum PT100 sensörü	69		
Üst konum ısı durumu	70		
Orta konum ısı durumu	71		
Alt konum ısı durumu	72		
Üst fan motoru durumu	73		
Alt fan motoru durumu	74		
Zincir kontrol sensörü	75		
Zincir sürücü durumu	76		
Zincir motor durumu	77		
Zincir mekanik durumu	78		

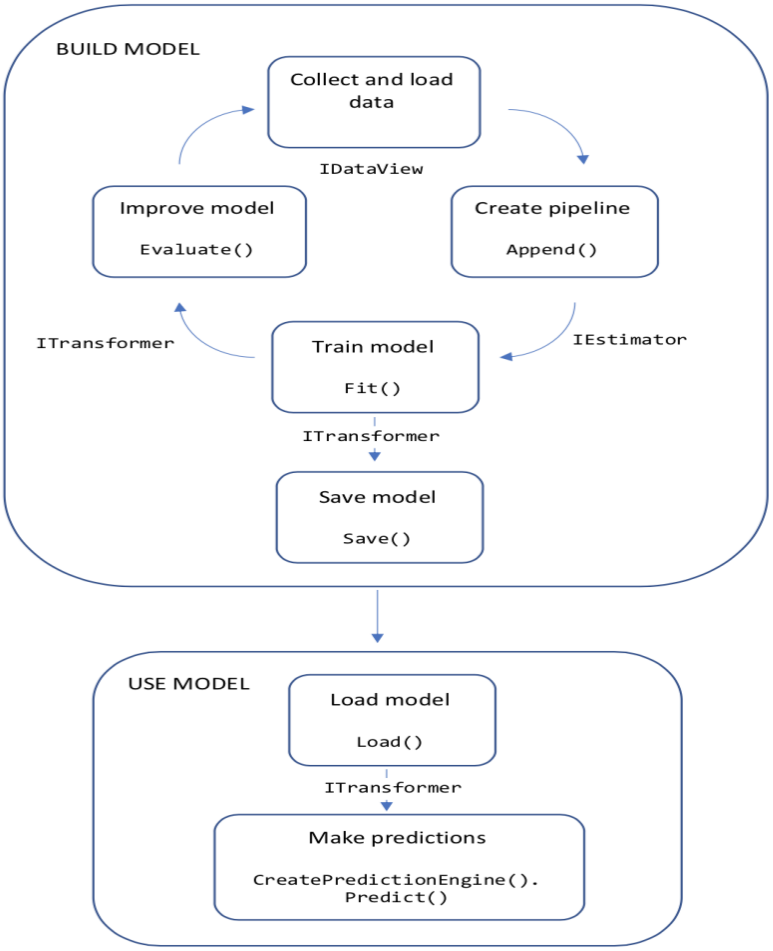
3.3.3. Form uygulaması veri tabanı entegrasyonu

Bu çalışmada, veri seti form uygulaması üzerinden arıza durumları için giriş yapılan senaryolar MSSQL veri tabanı üzerindeki veri seti tablosuna kaydedilmektedir. Veri tabanındaki tablo 80 kolondan oluşmaktadır. Bu kolonlardan 79 tanesi kontrol noktalarını, 1 tanesi ise arıza kodunu tanımlamak içindir. Form üzerinden her yapılan kayıt esnasında önce tablodaki tüm satırlar okunup kaydedilmek istenen tüm kontrol noktalarıyla karşılaştırılıp, eğer benzer bir satır kayıt var ise sistem tarafından kayıt yapılmamaktadır. Bu sayede veri setinde aynı kayıttan birden fazla bulunması ihtimalinin önüne geçilmiştir.

Bu tez çalışmasında, sıcak hava fırının transfer,duşlama ve tünel bölgelerine ait toplamda 1909 satırlık bir veri seti oluşturulmuştur. Daha sonra ana veri seti üzerinden bölgelere ait kayıtlar ayrıştırılarak transfer bölgesi için öğrenme algoritması kullanılmıştır.

3.4. Microsoft Machine Learning Framework (ML.NET 1.50)

ML.NET, çevrimiçi veya çevrimdışı senaryolarda .NET uygulamalarına makine öğrenimi ekleme olanağı sunmaktadır. Merkezi ML.NET, bir makine öğrenimi modelidir. Model, giriş verilerini bir tahmine dönüştürmek için gereken adımları belirtmektedir. ML.NET ile, bir algoritma belirterek özel bir model eğitilebilmektedir. Sınıflandıma işlemleri de ML.NET ile modellenebilmektedir. Aşağıda ML.NET çalışma modeli gösterilmiştir (Anonim, 2019a).



Şekil 3.3. ML.NET çalışma modeli (Anonim, 2019a)

Bu tez çalışmasında, veri setini eğitmek için ML.NET kütüphanesi kullanılmıştır.

3.5. ML.NET 1.50 Multiclass Classification Trainer Sınıfı

Bu sınıf, çoklu sınıflandırma eğiticilerini içermektedir. Bu sınıf içerisinden "AveragedPerceptronOva", "FastForestOva" ve "LightGbm" eğiticileri bu tez çalışmasında veri setini eğitmek için kullanılmıştır.

3.5.1. Averaged perceptron ova öğrenme metodu

Perceptron temelli bir algoritma içermektedir. Ayırıcı bir hiper düzlem bularak tahminlerini yapan bir sınıflandırma algoritmasıdır. Bir çevrimiçi algoritmadır, yani eğitim setindeki örnekleri birer birer işlemektedir. Bir dizi başlangıç ağırlığıyla başlar (sıfır, rastgele veya önceki bir öğreniciden başlatılır). Daha sonra, eğitim setindeki her örnek için özelliklerin ağırlıklı toplamı hesaplanır. Bu değer geçerli örneğin etiketiyle aynı işarete sahipse, ağırlıklar aynı kalır. Zıt işaretlere sahiplerse, ağırlık vektörü, mevcut örneğin özellik vektörü eklenerek veya çıkarılarak (sırasıyla etiket pozitif veya negatif ise), öğrenme oranı olarak adlandırılan $0 < a \leq 1$ faktörüyle çarpılarak güncellenir. Bu algoritmanın genelleştirilmesinde ağırlıklar, özellik vektörünün öğrenme hızıyla çarpılması ve bazı kayıp fonksiyonlarının gradyanı, sıfırdan farklıdır (Anonim, 2019b).

Averaged Perceptron'da her bir tekrarlama için, yani eğitim verisinden geçmek için, bir ağırlık vektörü yukarıda açıklandığı gibi hesaplanmaktadır. Son tahmin daha sonra her ağırlık vektöründen ağırlıklı toplamın ortalaması alınarak ve sonucun işaretine bakarak hesaplanmaktadır (Anonim, 2019b).

3.5.2. Fast forest ova öğrenme metodu

Karar ağaçları, girdiler üzerinde bir dizi basit test gerçekleştiren parametrik olmayan modellerdir. Bu karar prosedürü girişleri, işlenen örneğe benzeyen eğitim veri kümesinde bulunan çıktılarla eşleştirir. İkili ağaç veri yapısının her bir düğümünde, uygun yaprak düğümüne ulaşılana ve çıktı kararı geri dönene kadar her bir örneği ağacın dalları boyunca yinelemeli olarak eşleyen bir benzerlik ölçüsüne dayalı olarak bir karar verilmektedir (Anonim, 2019c).

Karar ağaçlarının çeşitli avantajları vardır:

- Eğitim ve tahmin sırasında hem hesaplama hem de bellek kullanımında etkilidirler.

- Doğrusal olmayan karar sınırlarını temsil edebilirler.
- Entegre özellik seçimi ve sınıflandırması yaparlar.
- Gürültülü özelliklerin varlığında dayanıklıdırlar.

Fast Forest rastgele bir orman uygulamasıdır. Model, karar ağaçları topluluğundan oluşmaktadır. Karar ormanındaki her ağaç tahmin yoluyla bir Gauss dağılımı vermektedir. Modeldeki tüm ağaçlar için kombine dağılıma en yakın Gauss dağılımını bulmak için ağaç topluluğu üzerinde bir toplama yapılmaktadır. Bu karar orman sınıflandırıcısı, karar ağaçlarından oluşan bir topluluktan oluşmaktadır (Anonim, 2019).

Genellikle, topluluk modelleri tek karar ağaçlarından daha iyi kapsama alanı ve doğruluk sağlamaktadır. Karar ormanındaki her ağaç bir Gauss dağılımı vermektedir (Anonim, 2019c).

3.5.3. Light gbm öğrenme metodu

LightGBM, gradyan artırıcı karar ağacının açık kaynaklı bir uygulamasıdır. Aşağıdaki avantajlarla verimli olacak şekilde tasarlanmıştır :

- Daha hızlı egzersiz hızı ve daha yüksek verimlilik,
- Daha az bellek kullanımı,
- Daha iyi doğruluk,
- Paralel ve GPU öğreniminin desteklenmesi,
- Büyük ölçekli verileri işleyebilmektedir.

Karar ağacı öğrenimi için ön sıralama tabanlı algoritmalar kullanmaktadır. Basit bir çözümdür ancak optimize edilmesi kolay değildir. LightGBM, sürekli özellik (öznitelik)

değerlerini ayrı bölmelere koyan histogram tabanlı algoritmalar kullanmaktadır. Bu, eğitimi hızlandırmakta ve bellek kullanımını azaltmaktadır (Anonim, 2019d).

LightGBM çeşitli paralel algoritmaların kullanımlarını desteklemektedir. Büyük veri setlerinde verileri paralel kullanmak hız açısından önemlidir. Çoğu karar ağacı büyüme olarak ağacın derinliği yönünde büyüme algoritması kullanmaktadır. LightGBM ise büyümek için maksimum delta kaybı olan yaprağı seçen bir algoritma kullanmaktadır. Bu algoritma derinlik yönünde büyüme algoritmasına göre daha düşük kayıp sağlama eğilimindedir (Anonim, 2019d).

4. BULGULAR ve TARTIŞMA

4.1. Bulgular

Bu tez çalışmasında oluşturulan eğitim seti verilerinden sıcak hava fırını transfer bölgesine ait olan veri seti ML.NET kütüphanesi aracılığı ile "AveragedPerceptronOVA", "FastForestOVA", "LightGBM" eğiticileri ile eğitilmiştir. Bu eğitime ait sırasıyla aşağıdaki adımlar izlenmiştir:

- ML.NET üzerinden eğitim senaryosu seçilmiştir. Bu tez çalışmasına ait eğitim senaryosu sınıflandırma problemidir.
- Eğitim senaryosu seçiminden sonra MSSQL veri tabanı bağlantısı sağlanmıştır.
- Veri tabanı üzerinde bulunan eğitim verilerine ait tablo seçilmiştir.
- Seçilen tabloya ait verilerin yazılıma entegrasyonu sağlanmıştır.
- İlk olarak "AveragedPerceptronOVA" eğiticisi üzerinden eğitim işlemi gerçekleştirilmiştir.
- "AveragedPerceptronOVA" eğiticisi için mikro ve makro doğruluk oranları hesaplanmıştır. Eğitim süresi kaydedilmiştir.
- İkinci olarak "FastForestOVA" eğiticisi üzerinden eğitim işlemi gerçekleştirilmiştir.
- "FastForestOVA" eğiticisi için mikro ve makro doğruluk oranları hesaplanmıştır. Eğitim süresi kaydedilmiştir.
- Son olarak "LightGBM" eğiticisi üzerinden eğitim işlemi gerçekleştirilmiştir.
- "LightGBM" eğiticisi için mikro ve makro doğruluk oranları hesaplanmıştır. Eğitim süresi kaydedilmiştir.

Veri seti için ayrıca bir normalizasyon işlemi yapılmamıştır. Çünkü mevcut kütüphane içerisindeki FastForestOVA ve LightGBM eğiticileri için normalizasyon işlemine gerek duyulmamaktadır. AveragedPerceptronOVA eğiticisi için ise normalizasyon işlemi otomatik olarak metot içerisinde gerçekleştirilmiştir.

Elde edilen doğruluk oranları ve eğitim süreleri bir sonraki bölümde tablo halinde verilmiştir. Başarım oranlarına ve sürelerine göre en iyi performans gösteren algoritma Data Set oluşturmak için kullanılan form uygulamasına entegre edilmiştir. Bu sayede istenen bir senaryo oluşturulup yapay zeka yönteminin verdiği çıktı bu form uygulaması üzrerinde gözlemlenebilmiştir. Başarım oranlarına ait tablonun yanı sıra en performanslı algoritmaya ait örnek test sonuçları (form uygulaması üzerinde oluşturulan arıza senaryo durumları) bir sonraki bölümde gösterilmiştir.

4.2. Öğrenme Metotlarının Örnek Makine Modeli Üzerindeki Başarımları

Eğitim işlemlerinin ardından metotların başarımlarını kıyaslamak amacıyla her metot için mikro ortalama ve makro ortalama yöntemine göre doğruluk oranları hesaplanmıştır. Bunun yanı sıra eğitim süreleri de kıyaslamada kullanılmıştır.

Mikro ortalama doğru tahmin edilen örneklerin oranıdır. Makro ortalama, sınıf düzeyindeki ortalama doğruluktur. Her sınıfın doğruluğu hesaplanır ve makro doğruluk bu doğrulukların ortalamasıdır.

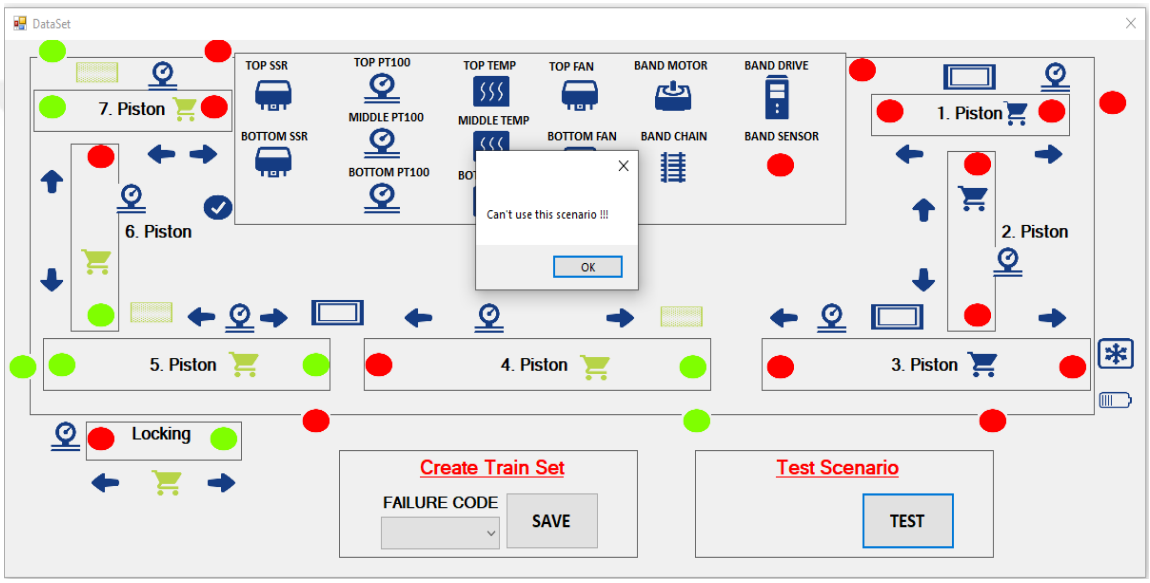
Bu tez çalışmasında kullanılan perceptron, rastgele orman ve gradient boosting temelli öğrenme algoritmalarına ait mikro, makro doğruluk ve eğitim süreleri aşağıda gösterilmiştir.

Çizelge 4.1. Öğrenme metotlarının karşılaştırması

	Mikro Doğruluk Oranı	**Makro Doğruluk Oranı**	**Eğitim Süresi**
AveragedPerceptronOVA	0.6454	0.5488	14.4 sn
FastForestOVA	0.8521	0.7388	71.0 sn
LightGBM	0.8732	0.7628	15.1 sn

Bunun yanı sıra en optimum olan LighGBM metodu Data Set form uygulamasına entegre edilmiştir. Bu sayede bir test senaryosu oluşturulup metot denenebilmektedir.

Bu entegrasyon yapılırken oluşturulacak senaryonun daha önce eğitimde kullanılan veri setinden bir senaryo olmaması için (eğer böyle bir durum olursa zaten bilinen bir senaryo için metodun sonucu yüzde yüz doğru olacaktır.) form uygulamasına bir algoritma yazılarak oluşturulan senaryo ile önce eğitim veri setinde olup olmadığı kontrolü yapılmaktadır. Eğer eğitim setine ait bir senaryo ise bir mesaj ile kullanıcı uyarılmaktadır. Eğer eğitim setine ait değilse o zaman LightGBM metotu tarafından senaryo değerlendirilip sonuç arıza kodu ekranda gösterilmektedir. Şekil 4.1. de daha önce veri setine tanımlanmış bir senaryonun test edilmeye çalışıldığındaki durumu verilmiştir.

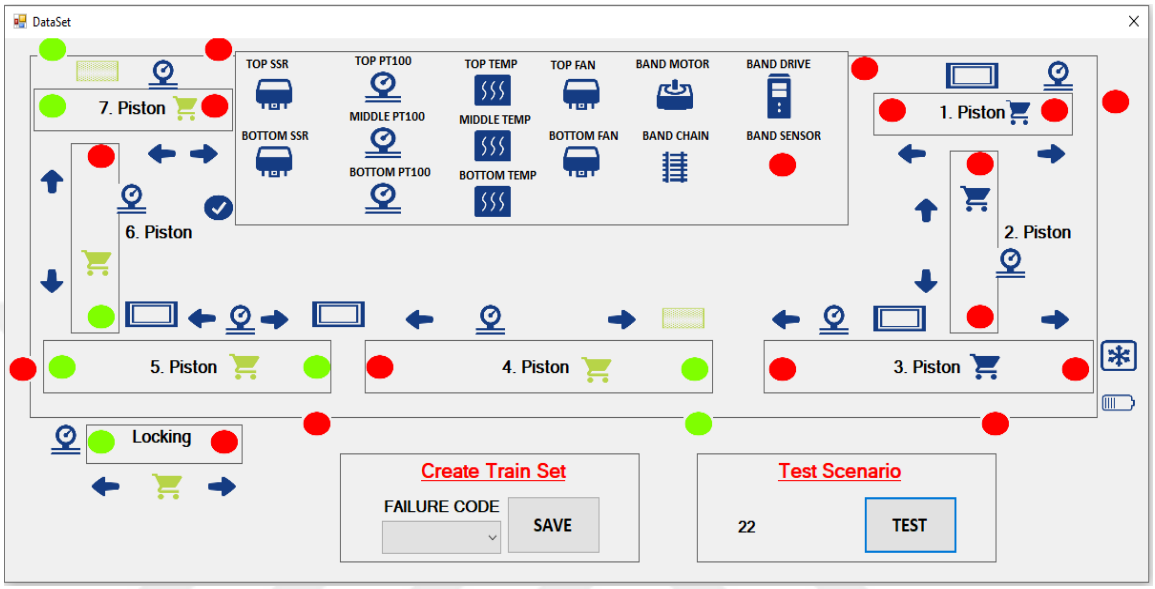


Şekil 4.1. Olumsuz test senaryosu

Şekil 4.1. de görüldüğü gibi uygun olmayan bir test senaryosu denenmek istenmiş ve bu senaryonun kullanılamayacağı bilgisi kullanıcıya verilmiştir.

Daha sonra 4. piston hava kaçağı arızası durumu test senaryosu için hazırlanmıştır. 4. piston üzerinde bir kalıp varken hava kaçağı yüzünden pistonun kalıbı götüremediği bu senaryoda aynı anda üzerinde kalıp olmayan 5. pistonun ileri konum sensörünün kalıp olmadığı halde sürekli çıkış verdiği bir durum kurgulanmıştır. Bu senaryoda iki durumlu bir arıza olacaktır. Ancak bakım ekibi açısından öncelikli ve önemli arıza durumu, üzerinde kalıp olan pistonun arızasıdır. Beklentimiz bu iki hata mevcut iken yapay zeka

metotunun “4. Piston Hava Kaçağı” arıza kodunu seçmesidir. Şekil 4.2. de kurgulanan senaryonun sonucu gösterilmiştir.



Şekil 4.2. Test senaryosu

Şekil 4.2. de görüldüğü üzere LightGBM eğitim modeli senaryoyu değerlendirerek “22” nolu arıza kodunu sonuç olarak döndürmüştür. Çizelge 3.1. de gösterildiği üzere “22” nolu arıza kodu “4. Piston Hava Kaçağı” arızasıdır. 5. Piston ileri konum sensöründe ki arızaya rağmen öğrenme metodu bize “22” nolu sonucu döndürmüştür. Bu bizim için önceliklendirme açısından doğru bir sonuçtur.

Burada eğitim setinde yapılan önceliklendirmeler doğrultusunda gerçekleştirilen eğitimin önemi gözükmektedir.

4.3. Tartışma

Bu tez çalışmasında yukarıda kıyaslamaları verilen öğrenme metotlarıyla yapılan eğitim sonucu başarım oranlarına bakıldığında, gradyan arttırıcı karar ağacı temelli olan geliştirilmiş “LightGBM” mikro ve makro doğruluk oranları açısından perceptron temelli

geliştirilmiş “AveragedPerceptronOVA” öğrenme metoduna göre yaklaşık olarak % 13 (Mikro Doğruluk Oranı) - % 22 (Makro Doğruluk Oranı) oranında daha doğru sonuç verdiği, rastgele orman temelli geliştirilmiş “FastForestOVA” öğrenme metoduna göre ise başarım oranlarının oldukça yakın olduğu görülmektedir. “FastForestOVA” öğrenme metodunun “AveragedPerceptronOVA” öğrenme metoduna göre yaklaşık olarak % 11 (Mikro Doğruluk Oranı) - % 19 (Makro Doğruluk Oranı) oranında daha doğru sonuç verdiği görülmektedir.

Öğrenme metotlarının eğitim sürelerine bakıldığında ise “LightGBM” öğrenme metodunun “FastForestOVA” öğrenme metoduna göre yaklaşık 5 kat daha hızlı çalıştığı, “AveragedPerceptronOVA” öğrenme metoduna göre ise yaklaşık olarak aynı sürelerde çalıştığı görülmektedir. “AveragedPerceptronOVA” öğrenme metodunun da “FastForestOVA” öğrenme metoduna göre yaklaşık 5 kat daha hızlı çalıştığı görülmektedir.

Öğrenme metotlarından tüm kıyaslama araçları baz alındığında örnek otomasyonel sistem için en ideal metodun “LightGBM” öğrenme metodu olduğu görülmektedir.

5. SONUÇ

Bu tez çalışmasında üç farklı temele dayalı yapay zeka metotlarıyla endüstriyel bir otomasyon makinesi üzerindeki arıza tespitine yönelik yapılan çalışma sonucunda bu makine için en ideal öğrenme metodunun bu üç yöntem içerisinden gradyan arttırıcı karar ağacı temelli metodun olduğu görülmüştür. Genellikle anlaşılması kolay ancak fazla zaman alıp daha düşük doğruluk oranları veren karar ağaçları algoritmasının bu geliştirilmiş şeklinin hem daha hızlı hem de daha doğru sonuçlar verdiği tespit edilmiştir.

Sonuç olarak bu tez çalışmasında endüstriyel bir otomasyon makinesi üzerinde doğru analizler sonucu oluşturulan uygun bir veri seti üzerinden makine, arıza ve ölçüm noktalarına bağlı olarak hız ve doğruluk oranları açısından uygun bir yapay zeka yöntemi ile arıza tespitlerinin yapılmasının uygulanabilirliliği gösterilmiştir.

Başka bir endüstriyel otomasyon makinesi için yapılacak uygulamalarda veri seti oluşturulurken veri setinin uygulanacak eğitim modelini kararsızlığa düşürmemesine dikkat edilmesi gerekmektedir. Uygulamanın yapılacağı endüstriyel sistem üzerinde rastlanılan arızaların birbirleriyle ilişkileri incelenmeli ve arıza önceliklerinin önem sıralarına göre bir eğitim seti oluşturulmalıdır. Diğer durumda eğitim başarım oranı ne denli yüksek olursa olsun birbiriyle ilişkili bir ya da daha fazla arıza durumunda yapay zeka sisteminin o an seçeceği arıza doğru da olsa önem olarak gerilerdeyse hedeflenen amaçtan uzak bir sonuç elde edilmiş olacaktır. Bu nedenle arızaların oluşma durumunda ölçüm alınacak noktalar ve bunların birbirleriyle ilişkileri detaylı olarak analiz edilmelidir. Bu analizin sonucunda, önem sıralarına göre bir eğitim seti oluşturulması faydalı olacaktır.

Bununla birlikte makine üzerinde arıza açısından bağımsız bölgeler oluşturarak, bu bölgeler üzerinde ayrı veri setleri üzerinden bir eğitim kurgusu kurulması da sistemin o an için karar alma sürecinde daha performanslı sonuçlar üretmesini sağlamaktadır. Ancak bu bağımsız bölgelerdeki ölçüm alınan noktaların azlığından kaynaklı olarak, ilgili bölgede oluşturulacak küçük boyutlu bir veri seti ise öğrenme metotları için yeterli olmayacaktır. Bu nedenle veri seti oluşturulurken makinedeki en ideal bölgelerin

seçilmesi önemlidir. Bununla birlikte, ölçüm noktalarının tespitinde seçilecek noktalarının ayırt edici özellikler taşıyor olması önemlidir. Benzer sonuçlar ya da birden fazla arızanın meydana gelmesi durumunda, ilişkili arızalar için benzer ölçüm noktalarının baz alınması sistem açısından yanıltıcı sonuçlar doğurabilir.

Doğru bölgeler oluşturulup doğru ölçüm noktalarından alınan verilerle oluşturulmuş bir eğitim seti elde edildiğinde, çeşitli öğrenme metotları kullanılarak makine üzerindeki arıza tespitinin yapay zeka kontrollü olarak yapılması mümkün olacaktır. Böylece arızaya müdahale edecek bakım operatörü yapay zeka sonucuna göre hızlı bir şekilde arızaya müdahale edebilecektir. Hatta sistem bir adım ileriye taşınarak oluşan arıza tespiti yanlış ise ikinci en yüksek tahmin seçilerek bakım operatörü yönlendirmesi yapılabilir. Doğru sonuç yine bulunamamışsa bir geri dönüş algoritmasıyla doğru arıza sonucu yapay zekaya öğretilerek sistem devamlılığı sağlanabilir, eğitim dinamik hale getirilebilir.

Yapılan bu tez çalışmasında, gerçek bir endüstriyel makine üzerinden alınan arıza durumlarına ait veriler simule edilerek kullanılan yapay zeka metotları kıyaslanmış ve başarı oranları tespit edilmiştir. Bu çalışma sonucunda, makine üzerinden toplanan gerçek verilerle sisteme en performanslı yapay zeka metodu uygulanarak arıza bakım sürelerinin % 50 oranında azalacağı, aynı arızanın tekrar edilebilirliliğinin % 80 oranında azalacağı ön görülmektedir.

KAYNAKLAR

AITopics, 2020. A Brief History of AI. https://aitopics.org/misc/brief-history (Erişim tarihi: 24.01.2020).

Alpaydın, E. 2009. Introduction to Machine Learning (2nd ed). Cambridge : The MIT Press.

Altunkaynak B. 2019. Veri Madenciliği Yöntemleri ve R Uygulamaları (2.Baskı), Ankara: Seçkin Yayıncılık.

Amit Y., Geman D. 1997. The Random Subspace Method for Constructing Decision Forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9 (7): 1545–1588.

Anonim, 2019a. ML.NET nedir ve nasıl çalışır? https://docs.microsoft.com/tr-tr/dotnet/machine-learning/how-does-mldotnet-work (Erişim tarihi: 31.12.2019).

Anonim, 2019b. Averaged Perceptron Trainer Class https://docs.microsoft.com/tr-tr/dotnet/api/microsoft.ml.trainers.averagedperceptrontrainer?view=ml-dotnet&viewFallbackFrom=ml-dotnet-preview (Erişim tarihi: 31.12.2019).

Anonim, 2019c. Fast Forest Binary Trainer Class https://docs.microsoft.com/tr-tr/dotnet/api/microsoft.ml.trainers.fasttree.fastforestbinarytrainer?view=ml-dotnet&viewFallbackFrom=ml-dotnet-preview (Erişim tarihi: 31.12.2019).

Anonim, 2019d. LightGBM's documentation https://lightgbm.readthedocs.io/en/latest/index.html (Erişim tarihi: 31.12.2019).

Anonim, 2019e. İTÜ elektrik arızasına yapay zeka ile önlem alacak. http://www.hurriyet.com.tr/ekonomi/itu-elektrik-arizasina-yapay-zeka-ile-onlem-alacak-41146248 (Erişim tarihi: 24.01.2020).

Balaban Erdal M., Kartal E. 2018. Veri Madenciliği ve Makine Öğrenmesi Temel Algoritmaları ve R Dili ile Uygulamaları (İkinci Baskı), İstanbul: Çağlayan Kitabevi ve Eğitim Çözümleri Ticaret A.Ş.

Bhatia, N., Vandana 2010. Survey of nearest neighbor techniques. *International Journal of Computer Science and Information Security*, 8 (2): 302–305.

Bih, J. 2006. Paradigm shift – an introduction to fuzzy logic. *IEEE Potentials*, 25 (1): 6–21.

Bishop, Christopher M. 2007. Pattern Recognition and Machine Learning. Singapore : Springer.

Bolat, B., Erol, K. O., İmrak, C. E. 2004. Genetic algorithms in engineering applications and the Function of operators. *Mühendislik ve Fen Bilimleri Dergisi*, 4 : 264–271.

Breiman, L. 1997. Arcing The Edge. *Technical Report 486.* Statistics Department, University of California, Berkeley.

Breiman L 2001. Random Forests. *Machine Learning,* 45 (1): 5–32.

Bridge, D. 2013. Classification : k Nearest Neighbours. http://www.cs.ucc.ie/~dgb/courses/tai/notes/handout4.pdf (Erişim tarihi: 25.01.2020).
Buchanan, B. G. 2005. A (Very) Brief History of Artificial Intelligence. *AI Magazine,* 26 (4): 53.

Camastra, F., Vinciarelli, A. 2008. Machine Learning for Audio, Image and Video Analysis : Theory and Applications. *London : Springer*.

Carbonell, J. G., Michalski, R. S., Mitchell, T. M. 1983. An overview of machine learning. R.S. Michalski, J.G. Carbonell, T.M. Mitchell(Ed.), Machine Learning An Artificial Intelligence Approach içinde (3-23 pp.) *Berlin, Heidelberg : Springer*.
Cheng Li 2019. A Gentle Introduction to Gradient Boosting.

http://www.chengli.io/tutorials/gradient_boosting.pdf (Erişim tarihi: 30.12.2019).
Chua L.O., Yang L., 1988. Cellular neural networks: Theory. *IEEE Transactions on Circuits and Systems*,35: 1257-1272.

Chua L.O., Yang L., 1988. Cellular neural networks: Theory. *IEEE Transactions on Circuits and Systems*,35: (10), 1273-1290.

Cuningham, P., Delany, S. J. 2007. K-Nearest neighbour classifiers. *Multiple Classifier Systems,* 1–17.

Dietterich, Thomas 2000. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning,* 40 (2): 139–157.

Duda, R. O., Hart, P. E., Stork, D. G. 2001. Pattern classification (2nd ed). *New York, NY, USA : John Wiley & Sons*.

Dominguez-Castro R., Espejo S., Rodriguez-Vazquez A., Carmona R., 1994. A cnn universal chip in cmos technology. *IEEE International Workshop CNNA*, 91-96.

Dopico, M., Gomez, A., De la Fuente, D., García, N., Rosillo, R., Puche, J. 2016. A vision of industry 4.0 from an artificial intelligence point of view. *Int'l Conf. Artificial Intelligence* ICAI'16, 407-413.

Frank, E., Hall, M., Pfahringer, B. 2002. Locally weighted naive bayes. *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence,* 249–256.

Friedman, J. H. 1999. Greedy Function Approximation: A Gradient Boosting Machine. https://statweb.stanford.edu/~jhf/ftp/trebst.pdf (Erişim tarihi: 30.12.2019).

Genevieve B. Orr 2019. Willamette University. https://www.willamette.edu/~gorr/classes/cs449/Classification/perceptron.html; (Erişim tarihi: 29.12.2019).

Geurts P, Ernst D, Wehenkel L 2006. Extremely Randomized Trees. *Machine Learning.* 63: 3–42.

Gou, J., Xiong, T., Kuang, Y. 2011. A novel weighted voting for k-nearest neighbor rule. *Journal of Computers,* 6 (5): 833–840.

Görz, G., Nebel, B. 2002. Yapay Zeka. İstanbul : İnkılap Kitabevi.

Gürsakal, N. 2017. Makine Öğrenmesi ve Derin Öğrenme. Bursa : Dora Yayıncılık.

Hastie, T., Tibshirani, R., Friedman, J. 2008. The Elements of Statistical Learning : Data Mining, Inference, and Prediction (2nd ed). New York : Springer.

Hastie Trevor, Tibshirani Rober, Friedman Jerome 2008. The Elements of Statistical Learning (2nd ed.), 587-588 pp.

Herbrich, R. 2002. Learning kernel classifiers : theory and alghoritms. *Cambridge, MA,* USA : The MIT Press.

Ho, Tin Kam 1995. Random Decision Forests America. *Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC,* 1995, 278–282.

Ho, Tin Kam 2002. A Data Complexity Analysis of Comparative Advantages of Decision Forest Constructors. *Pattern Analysis and Applications.* 5 (2): 102–112.

Ho TK 1998. Random Decision Forests America. Shape quanziation and recognition with randomized trees. *Neural Computation,* 20 (8): 832–844.

Jay Lee, Hossein Davari, Jaskaran Singh, Vibhor Pandhare 2018. Industrial Artificial Intelligence for industry 4.0-based manufacturing systems. *Manufacturing Letters (2018),* 18:20-23.

Kantrowitz, M. 1994. Milestones in the Development in Artificial Intelligence. http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/util/areas/faq/timeline.txt (Erişim tarihi: 24.01.2020).

Kulkarni, S., Harman, G. 2011. An Elemantary Introduction to Statistical Learning Theory (1th ed). Hoboken, New Jersey : John Wiley & Sons, Inc.

Mason, L., Baxter, J., Bartlett, P. L., Frean, Marcus 1999. Boosting Alghoritms as Gradient Descent, In S.A. Solla and T.K. Leen and K. Müller (ed.). Advances in Neural Information Processing Systems 12. MIT Press. 512–518 pp.

McCallum, A., Nigam, K. 1998. A comparison of event models for naive bayes text classification. AAAI-98 Workshop on Learning for Text Categorization, 41–48.

McCulloch, W. S., Pitts, W. A. 1943. A logical calculus of the ideas immanent in nervous activity. Buttetin of Mathematics and Biophysics, 5: 115-133.

McNamara, J. M., Green, R. F., Olsson, O. 2006. Bayes' theorem and its applications in animal behaviour. *Oikos*, 112 (2): 243–251.

Michael Haenlein and Andreas Kaplan 2019. A Brief History of Artificial Intelligence: On the Past, Present, and Future of Artificial Intelligence California Management Review 2019, Vol. 61: (4) 5–14.

Miller, S. 2011. Computer Scientist Coined "Artificial Intelligence. https://www.wsj.com/articles/SB10001424052970203911804576653530510986612 (Erişim tarihi: 30.12.2019).

Minsky, M., Papert, S. 1969. Perceptrons. MIT Press Cambridge, MA, 258 pp.

Mitchell, T. M. 1997. Machine Learning (1th ed). USA : The McGraw-Hill Companies, Inc.

Murphy, K. P. 2012. Machine Learning : A Probabilistic Perspective. Cambridge, MA : The MIT Press.

Nabiyev, V. V. 2005. Yapay Zeka. *Ankara :* Seçkin Yayıncılık.

Nabiyev, V. V. 2016. Yapay Zeka: Stratejili Oyunlar - Örüntülü Tanıma - Doğal Dil İşleme (5.Baskı) Ankara, Seçkin Yayıncılık.

Nilsson, N. J. 2009. The quest for artificial intelligence : a history of ideas and achievements. Cambridge, New York : Cambridge University Press.

Olazaran, Mikel 1996. A Sociological Study of the Official History of the Perceptrons Controversy. Social Studies of Science, 26 (3): 611–659.

Özen, Z., Kartal, E., Gülseçen, S. 2017. Yapay Zeka ve Yapay Sinir Ağları. T. R. Çölkesen, O. Aliefendioğlu (Ed.), Bilgisayar Bilimine Giriş içinde (1. Baskı., 523-558 s.). İstanbul : Papatya Bilim Kitabevi.

Özkan, Y. 2008. Veri madenciliği yöntemleri. Papatya Yayıncılık Eğitim.

Öztemel E. 2006. Yapay sinir ağları. İstanbul : Papatya Yayıncılık Eğitim.

Rai, P. 2011. Model Selection and Feature Selection http://www.cs.utah.edu/~piyush/teaching/22-9-print.pdf (Erişim tarihi: 24.01.2020).

Rodriguez-Vazquez A., 2004. ACE16k: The third generation of mixed-signal SIMD-CNN-ACE chip toward VSoCs. . IEEE Transactions on Circuits and Systems Part I:Regular Paper, 51, (5), 851-863.

Rosenblatt, F. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. Psychoanalytic Review, 65: 386-408.

Rokach, L., Maimon O. 2005. Decision Trees. L. Rokach, O. Maimon (Ed.), Data mining and knowledge discovery handbook içinde (165-192 pp.). Boston, MA : Springer.

Roska T., Chua L.O. 1993. The CNN Universal Machine. *IEEE Transactions on Circuits and Systems*, 40: 163-173.

Rudin, C. 2012. K-NN. https://ocw.mit.edu/courses/sloan-school-of-management/15-097-prediction-machine-learning-and-statistics-spring-2012/lecture-notes/MIT15_097S12_lec06.pdf (Erişim tarihi: 24.01.2020).

Russel, S., Norvig, P. 2003. Artificial Intelligence : A Modern Approach.(2nd ed.). USA: Prentice Hall.

Sayad, S 2015. Decision Tree Regression http://saedsayad.com/decision_tree_reg.htm (Erişim tarihi: 24.01.2020).

Sarı, M., Murat, Y., Kırbalı, M. 2005. Fuzzy Modelling Approach and Applications. *Journal of Natural and Applied Scienes Institute of Dumlupınar University,* (9): 77–92.

Steinbach, M., Tan, P. N. 2009. kNN: k-Nearest Neighbors. X. Wu, V.Kumar(Ed.), The top ten algorithms in data mining içine (151-162 pp.). Boca Raton, FL, USA : Chapman and Hall / CRC Press.

Sydney, F. 2019. Seeing the Forest for the Trees: An Introduction to Random Forest, https://community.alteryx.com/t5/Alteryx-Designer-Knowledge-Base/Seeing-the-Forest-for-the-Trees-An-Introduction-to-Random-Forest/ta-p/158062 (Erişim tarihi: 24.01.2020).

Tuptuk N, Hailes S. 2018. Security of smart manufacturing systems. *J Manuf Syst (2018),* 47: 93-106.

Türk Dil Kurumu 2020. Zeka. Türk Dil Kurumu Güncel Türkçe Sözlük. https://sozluk.gov.tr/ (Erişim tarihi: 24.01.2020).

Veksler O. 2012. Artificial Intelligence II Lecture 3 : Machine Learning K Nearest Neighbor Classifier http://www.csd.uwo.ca/courses/CS4442b/L3-ML-knn.pdf (Erişim tarihi: 24.01.2020).

Venkat 2017. Classification Series 5 – K-Nearest Neighbors (knn). https://dslytics.wordpress.com/2017/11/16/classification-series-5-k-nearest-neighbors-knn/ (Erişim tarihi: 24.01.2020).

Verma, D., Kakkar, N., Mehan, N. 2014. Comparison of Brute-Force and K-D Tree Alghoritm**.** *International Journal of Advanced Research in Computer and Communication Engineering* , 3: (1) , 5291-5297.

Yıldız, A . 2018. Endüstri 4.0 ve akıllı fabrikalar**.** *Sakarya University Journal of Science* , 22 (2) , 546-556.

Zhang, K. 2006. Decision Tree Algorithm. http://www.cse.ust.hk/~twinsen/Decision_Tree.ppt (Erişim tarihi: 24.01.2020).

Zurada, J. M. 1992. Introduction to Artificial Neural Systems, West Publishing Company, USA, 758 pp.

ÖZGEÇMİŞ

Adı Soyadı : Oğuzhan ÇÖMLEKCİ
Doğum Yeri ve Tarihi : Bursa / 1991
Yabancı Dil : İngilizce

Eğitim Durumu (Kurum ve Bitirme Yılı)
Lise : Bursa Anadolu Erkek Lisesi, 2009
Lisans : Bursa Uludağ Üniversitesi, 2013

Çalıştığı Kurum/Kurumlar : İletişim Yazılım (2015-2018)
TI Fluid Systems (2018- ...)

İletişim (e-posta) : oguzhancomlekci0816@gmail.com