Meta-Sezgisel Yöntemler ile Müzik Verisi Üzerinde Özellik Seçimi ve Kategorizasyon

Abdurrahim Hüseyin Ezirmik

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Ocak 2020

Metaheuristic Methods For Feature Selection And Categorization On Music Data

Abdurrahim Hüseyin Ezirmik

MASTER OF SCIENCE THESIS

Department of Computer Engineering

January 2020

Meta-Sezgisel Yöntemler ile Müzik Verisi Üzerinde Özellik Seçimi ve Kategorizasyon

Abdurrahim Hüseyin Ezirmik

Eskişehir Osmangazi Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Bilimleri Bilim Dalında
YÜKSEK LİSANS TEZİ
Olarak Hazırlanmıştır

Danışman: Prof.Dr. İdris Dağ

Ocak 2020

ONAY

Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans öğrencisi Abdurrahim Hüseyin Ezirmik'in YÜKSEK LİSANS tezi olarak hazırladığı "Meta-Sezgisel Yöntemler ile Müzik Verisi Üzerinde Özellik Seçimi ve Kategorizasyon" başlıklı bu çalışma, jürimizce lisansüstü yönetmeliğin ilgili maddeleri uyarınca değerlendirilerek oy birliği ile kabul edilmiştir.

Danışman **:** Prof. Dr. İdris Dağ

İkinci Danışman **: -**

Yüksek Lisans Tez Savunma Jürisi:

Üye : Prof. Dr. İdris Dağ

Üye : Doç. Dr. Cüneyt Akınlar

Üye : Dr. Öğr. Üyesi Gültekin Kuvat

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ..... tarih ve ..... sayılı kararıyla onaylanmıştır.

Prof. Dr. Hürriyet ERŞAHAN
Enstitü Müdürü

ETİK BEYAN

Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kılavuzuna göre, Prof. Dr. İdris Dağ danışmanlığında hazırlamış olduğum "Meta-Sezgisel Yöntemler ile Müzik Verisi Üzerinde Özellik Seçimi ve Kategorizasyon" başlıklı tezimin özgün bir çalışma olduğunu; tez çalışmamın tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; tezimde verdiğim bilgileri, verileri akademik ve bilimsel etik ilke ve kurallara uygun olarak elde ettiğimi; tez çalışmamda yararlandığım eserlerin tümüne atıf yaptığımı ve kaynak gösterdiğimi ve bilgi, belge ve sonuçları bilimsel etik ilke ve kurallara göre sunduğumu beyan ederim. 06/01/2020

Abdurrahim Hüseyin Ezirmik

ÖZET

Günümüzde multimedya içerik üretimi yüksek boyutlara ulaşmıştır. Bu miktar artışı değerli içeriğe erişmenin zorlaşmasına sebep olmuştur. Veri madenciliği anlamlı veriye ulaşmak için gerekli hale gelmiştir. Madencilik sürecinin önemli bir adımı da veri boyutunun azaltılmasıdır. Özellik seçimi, veri kümesinde bulunan ilgisiz, gürültülü veya eksik verileri çıkarak veri boyutunu azaltır. Bu şekilde, veri analizinde kullanılan yöntemlerin daha verimli ve hızlı çalışmasını sağlar.

Bu tezde, doğadan esinlenen meta-sezgisel algoritmalar ve yapay sinir ağları kullanılarak özellik seçimi yapılmıştır. Seçilen özelliklerin bulunduğu özelleştirilmiş veriler birçok yöntem ile sınıflandırılmıştır. Bu çalışma, analiz edilen müzik veri setinde bazı iyileştirmeler yapılarak sınıflandırma başarımının artırılması konusuna odaklanmıştır. Kullanılan yöntemler karşılaştırılmalı olarak sunulmuş ve elde edilen sonuçlar değerlendirilmiştir.

Anahtar Kelimeler: Metasezgisel algoritmalar, Özellik seçimi, Veri madenciliği, Sınıflandırma metotları, Makine öğrenmesi

SUMMARY

Nowadays, multimedia content production has reached high levels. This amount increase made it difficult to access valuable content. Data mining has become necessary to reach meaningful data. An important step in the mining process is the reduction of the data size. The feature selection reduces the size of the data by removing unrelated, noisy or missing data from the data set. In this way, it enables the methods used in data analysis to work more efficiently and faster.

In this thesis, feature selection is made by using nature-inspired metaheuristic algorithms and artificial neural networks. Customized data with selected features are classified by many methods. This study focused on increasing the classification performance by making some improvements in the analyzed music dataset. The methods used are presented comparatively and the results obtained are evaluated.

Keywords: Metaheuristic algorithms, Feature selection, Data mining, Classification methods, Machine learning

TEŞEKKÜR

Tez çalışmamda bana yol gösteren saygıdeğer danışman hocam Prof. Dr. İdris Dağ'a, bu süreçte bana maddi ve manevi desteklerini sunan aileme ve yanımda olan dostlarıma teşekkür ederim.

İÇİNDEKİLER

Sayfa

İÇİNDEKİLER (devam)

Sayfa

ŞEKİLLER DİZİNİ

Şekil **Sayfa**

ÇİZELGELER DİZİNİ

Çizelge **Sayfa**

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar	Açıklama
KKA	Karınca Koloni Algoritması
PSO	Parçacık Sürü Optimizasyonu
BT	Benzetilmiş Tavlama
GA	Genetik Algoritma
KNN	K-En Yakın Komşuluk Sınıflandırıcısı
MLP	Çok Katmanlı Algılayıcı

1. GİRİŞ VE AMAÇ

Günümüz dünyasında müzik insan hayatının her alanında yer almaktadır. Sürekli büyüyen eğlence sektörünün de etkisiyle zamanla üretilen müzik verisinin boyutu artmakta ve böylece dinleyiciler bu verinin tamamına erişime konusunda yetersiz kalabilmektedir. Eğer müzik keşfetmek için iyi bir yöntem kullanılmazsa üretilen müziğin kayda değer bir bölümü gözden kaçabilmektedir. Multimedya içeriğin genişlemesi ve dijital kütüphanelerin de giderek artmasıyla birlikte bilgi edinim ve erişimi daha önemli bir hal almaktadır.

Bu çalışmada sezgisel algoritmalar kullanılarak şarkıları kategorize edebilen bir sistem tasarlamak amaçlanmıştır. Bir ses dosyasını incelemek için öncelikle verilen bilginin tipini belirlemek gerekir. Müzik, konuşma ve ses üzerine birçok araştırma yapılmıştır. Bunun yanı sıra şarkılar hakkında yapılan çalışmalar nispeten azdır ve halen devam etmektedir. Şarkılarla ilgili lirik, tür ve dönem gibi bilgiler internette paylaşılmaktadır. Dijital müzik sanatçı, parça adı, yıl gibi bilgiler için kaynak oluşturmaktadır. Bu bilgileri kullanarak birçok işlem yapılabilmektedir. Parça sınıflandırma ve şarkı öneri sistemleri bunlara örnektir.

Son zamanlarda, öznitelik seçimi araştırmaları çeşitli nedenlerle artış göstermiştir. Bunun nedeni, veri madenciliği, tıbbi veri işleme ve multimedya bilgi alma gibi büyük miktarda veri ile ilgilenen yeni uygulamalar geliştirilmiş olmasıdır. Öznitelik seçimi verimli ve yaygın bir şekilde sınıflandırma sistemlerinde kullanılmaktadır. Ayırt edici özniteliklerin bulunması, tanıma başarısını arttırmaktadır. Seçilen özniteliklerle yapılan sınıflandırmada işlem sayısı daha azdır, gürültülü ve ilgisiz öznitelikler özgün veriden çıkarılarak sınıflama başarısı arttırılır, öznitelikler üzerinden yapılabilen sınıflama yorumları artar veya kolaylaşır. Eğitim zamanı kısalır, daha az ölçüm yapılır ve daha az bellek kullanılır. Bunlar, anlamlı ve daha kolay sınıflandırma sağlar (Çetişli, 2006).

2. LİTERATÜR ARAŞTIRMASI

2.1. Meta-Sezgisel Yöntemler

Meta-sezgisel yöntem, farklı sezgisel optimizasyon algoritmalarının geliştirilmesine kılavuzluk eden veya strateji sağlayan, problemden bağımsız bir algoritmik çerçevedir. Bu terim ayrıca, bir sezgisel optimizasyon algoritmasının belirlenen soruna spesifik bir şekilde uygulanmasını ifade etmek için de kullanılır. Teknik olarak ilk defa 1986 yılında Fred Glover tarafından dile getirilen meta-sezgisel terimi Yunanca "meta" kelimesi ile "heuristic" kelimesinin birleşiminden oluşmakta ve üst seviye sezgisel anlamına gelmektedir.

Üst seviye sezgisel yaklaşım, çözüm uzayında olasılık temelli ancak bilinçli bir mantıkla arama gerçekleştiren yöntemleri içermektedir. Bu yöntemler her adımda oluşturulan çözüm kümesinden yola çıkarak yeni çözümler üretmektedirler. Böylece arama uzayının en uygununa yakın olan noktalarında aramalar yapılarak, yerel en iyi nokta seçiliminden de kurtularak en uygun çözüme ulaşmaya çalışılır.

Meta-sezgisel yöntemler arama işlemine yön veren metotlardır. Arama uzayını etkili bir şekilde keşfederek en iyi veya en iyiye en yakın sonuçları elde etmeyi amaçlamaktadırlar. Yerel arama tekniklerinden, karmaşık öğrenme işlemlerine kadar yayılım gösteren yapıdadırlar. Genelde belirleyici olmayan, yaklaşık bir çözüm sunan yöntemlerdir. Sadece belirlenmiş bir probleme değil farklı tiplerde problemlere çözüm getirirler. Arama uzayında yerel çözümlere yakınsamayı engelleyecek şekilde tasarlanmışlardır.

Meta sezgisel algoritmaların iyi sonuçlar üretebilmesi yöntemin temel kavramlarının probleme iyi bir şekilde adapte edilmesi ile mümkündür. Birçok meta-sezgisel algoritma çeşidi mevcuttur. Doğadan esinlenerek geliştirilen algoritmalar canlıların doğal ortamlardaki davranış biçimlerini taklit etmektedir. Popülasyon tabanlı veya bireysel olanları mevcuttur. Amaç fonksiyonları statik ya da dinamik olabilmektedir. Komşuluk durumlarına veya hafıza kullanıp kullanmadıklarına göre ayrılabilirler. Belirtilen yöntemler, klasik sezgisel algoritmaların doğadan esinlenerek geliştirilmiş halleri olarak görülmektedir. Yöntemlerin farklı bilim dalları

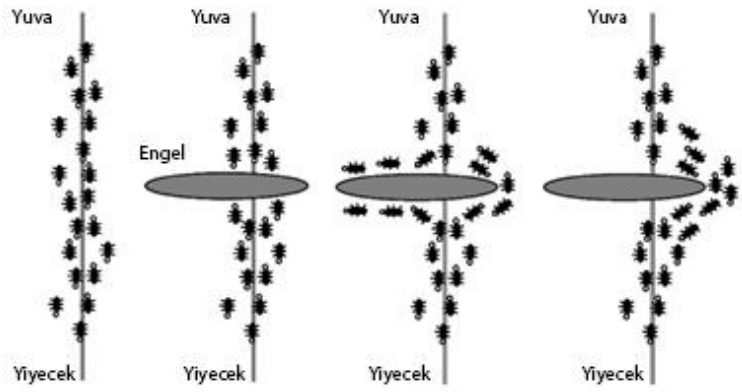
üzerinde temel alınarak optimizasyon amaçlı kullanım için geliştirilmesiyle algoritma çeşitliliği artırılmıştır.

2.1.1. Karınca Koloni Algoritması

Karınca Koloni Optimizasyonu, zor optimizasyon problemlerini çözmek için geliştirilen bir metasezgisel yöntemdir. Doğal ortamlarında salgıladıkları feromon hormonunu iletişim aracı olarak kullanan gerçek karıncaların davranışlarından ilham almıştır. Bu optimizasyon yöntemi biyolojik örneğe benzer şekilde, yapay karınca kolonisinde feromon izlerini kullanarak dolaylı yoldan kurulan iletişimi temel alır. Feromon, karıncaların bir probleme olasılıksal olarak çözümler üretmekte kullandıkları ve algoritmanın yürütülmesi sırasında adapte ettikleri, arama deneyimlerini yansıtmak için kullandıkları dağıtık, sayısal bir bilgi olarak işlev görür.

Bu algoritmanın ilk örneği, iyi bilinen gezgin satıcı problemini (GSP) çözmek amacıyla önerilen Karınca Sistemidir (KS). Başlangıçta çözümleri iyileştirmesine rağmen, GSP için geliştirilen son teknoloji algoritmalarla rekabet edememiştir. Bununla birlikte, hem çok daha iyi bir hesaplama performansı elde eden algoritma çeşitliliğinin artması hem de çok çeşitli farklı problemlerin uygulamaları için yapılacak daha fazla araştırmaları teşvik etme konusunda önemli rol oynamıştır. Bu yöntemi kullanarak ardışık sıralama, iş çizelgeleme, komünikasyon ağlarının belirlenmesi, grafik renklendirme gibi kayda değer sayıda uygulama alanında oldukça başarılı performans elde edilmiştir. Karınca koloni optimizasyonu (KKO) mevcut uygulamalar ve algoritma türevleri için genel bir çerçeve sağlar. Bu sezgisel yöntem kullanılarak geliştirilen algoritmaya ise Karınca Koloni Algoritması (KKA) adı verilir.

Gerçek karıncalar ortak davranışlarda bulunarak gıda kaynaklarına en kısa yolun bulunması ve ulaşılan gıdanın yuvaya taşınması gibi karmaşık görevleri yerine getirmektedirler. Karınca koloni algoritması basit bir iletişim mekanizması kullanarak bir karınca kolonisinin iki nokta arasındaki en kısa yolu bulabilmesi prensibini taklit eder. Görme yetisi iyi olmayan karıncaların oluşturduğu koloninin yuva ile gıda kaynağı arasında gidiş ve dönüş yapabildikleri bir yol vardır. Gezileri sırasında karıncalar yerde kimyasal bir iz (feromon) bırakır. Feromon kokusu olan ve uçucu bir maddedir. Bu iz diğer karıncaları hedef noktaya doğru yönlendirmede rol oynar. Belirli bir yoldaki feromon miktarı arttıkça, karıncaların o yolu seçme olasılığı da artar.



Şekil 2.1. Yiyecek ve yuva arasında en uygun yolu arayan bir karınca kolonisi (Talbi, 2009)

Ayrıca, bu kimyasal madde zaman içinde buharlaşarak azalan bir etkiye sahiptir ve bu maddenin bir karınca tarafından salgılanma miktarı, ortamdaki gıda miktarına bağlıdır. Şekil 2.1'de gösterildiği gibi, bir engelle karşı karşıya kaldığında her karıncanın sol veya sağ yolu seçmesi için eşit bir olasılık vardır. Sol iz sağdakinden kısa olduğundan ve daha az seyahat süresi gerektirdiğinden, karınca daha yüksek miktarda feromon bırakacaktır. Karıncalar bir yolu ne kadar çok kullanırsa o yolda biriken feromon izi de o kadar fazladır. Böylelikle, en kısa yol belirlenmiş olur.

```
Procedure Karınca_Koloni_Algoritması
Begin
        İlk feromon miktarının hesaplanması
        while(not durdurma_kriteri)
                Aday çözümler oluştur
                Lokal arama gerçekleştir
                Feromonları güncelle
        end while
        return bulunan en iyi çözüm
End
```

Üstteki algoritma yapısı KKO için bir şablon sunar. İlk olarak, feromon bilgisi girilir. Algoritma temel olarak çözüm oluşturma ve feromon güncelleme olarak yinelenen iki adımdan oluşur.

Çözümlerin oluşturulması, olası bir durum geçişi kuralına göre yapılır. Yapay karıncalar, tam bir çözüm elde edilinceye kadar kısmi olanlara çözüm bileşenleri ekleyerek, olasılıklı bir şekilde sonuca varan rastlantısal açgözlü işlemler olarak kabul edilebilir. Hedef

optimizasyon problemi, bir karıncanın yol inşa edeceği bir karar grafiği olarak görülebilir. Böylelikle, yinelemeli işlemler yapılarak optimal çözüme ulaşmak amaçlanır.

KKA'da geçiş kuralı belli bir olasılığa göre iki şekilde gerçekleştirilir. İlk seçenek q_0 olasılıkla feromonun en yoğun olduğu yolun seçilmesidir. q_0 parametresi genellikle % 90 olarak belirlenir. $\tau(i,j)$, i ve j noktaları arasındaki feromon miktarı, seçilebilirlik parametresi $\eta(i,u)$, i ve j noktaları arasındaki mesafenin tersi $(1/\delta(i,j))$, α ve β ayarlanabilir parametreler olmak üzere, i noktasında bulunan bir karıncanın gideceği nokta aşağıdaki gibi seçilmektedir:

$$j = \max_{u \in j_k(i)} \{[\tau(i,u)]^{\alpha} \times [\eta(i,u)]^{\beta}\} \quad eğer \;\; q \leq q_0 \tag{2.1}$$

İkinci seçenek ise gidilmesi mümkün olan yollardan birini, yollardaki feromon izleriyle orantılı olarak seçmektir. Bu şekilde yol seçimi olasılığı 1- q_0 oranındadır. $J_k(i)$, i noktasındaki karıncanın gidebileceği noktalar yani ziyaret edilmemiş şehirleri temsil eder. Tüm noktalar için seçilme olasılıkları aşağıdaki gibi hesaplanmaktadır:

$$p_k(i,j) = \begin{cases} \dfrac{[\tau(i,j)]^{\alpha} \times [\eta(i,j)]^{\beta}}{\sum_{u \in J_k(i)} [\tau(i,j)]^{\alpha} \times [\eta(i,j)]^{\beta}} & eğer \;\; j \in j_k(i) \\ 0 & diğer\ durumlarda \end{cases} \tag{2.2}$$

Feromon izleri, karıncalar tarafından yeni çözümlerin oluşturulmasını yönlendirecek olan iyi üretilmiş çözümlerin özelliklerini ezberler. Bu izler, edinilen bilgiyi yansıtmak için arama sırasında dinamik olarak değişir ve tüm karınca arama sürecinin hafızasını temsil eder. Probleme bağlı sezgisel bilgi, karıncalara çözüm üretmek için aldıkları kararlarında daha fazla ipucu verir.

Feromon güncellemesi, üretilen çözümler kullanılarak gerçekleştirilir. İlk olarak tüm yollardaki feromonlar, belirlenen oranda buharlaştırılmaktadır. Daha sonra karıncaların geçiş yaptığı yollardaki feromon miktarları, o yolu kullanan karıncanın toplam yol uzunluğuyla ters orantılı olarak arttırılmaktadır. Böylelikle daha kısa yola sahip karıncaların kullandıkları yollardaki feromon miktarları daha fazla artış göstermektedir. Lokal ve küresel olarak feromon güncellemesi yapabilmek mümkündür.

$\tau_{ij}(t)$, t iterasyonuna kadar biriken feromon düzeyi, $\Delta\tau_{ij}^{k}(t+1)$, t iterasyonundaki feromon düzeyi ve ρ, feromon buharlaşma parametresi olmak üzere lokal feromon düzeyi aşağıdaki formülle hesaplanır:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\,\tau_{ij}^{k}(t+1) \tag{2.3}$$

$$\Delta\tau_{ij}^{k}(t+1) = \begin{cases} 1/L^{k}\,(t+1) & k\ karıncası\ (i,j)\ yolunu\ kullanmışsa, \\ 0 & diğer\ durumlarda \end{cases} \tag{2.4}$$

$L^{k}(t+1)$ k karıncasının toplam tur uzunluğudur. Lokal feromon güncellemesi, turları dinamik olarak değiştirerek geçiş yapılan yolları cazip hale getirir. Karıncalar değişen feromon miktarlarına bağlı olarak her iterasyonda turlarını da değiştirmektedirler. Böylelikle sürekli olarak daha kısa turları bulmak amaçlanmaktadır.

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}^{k}(t+1) \tag{2.5}$$

$$\Delta\tau_{ij}^{k}(t+1) = \begin{cases} \frac{1}{L_{best}(t+1)} & (i,j)\ en\ iyi\ tura\ aitse, \\ 0 & diğer\ durumlarda \end{cases} \tag{2.6}$$

$L_{best}(t+1)$ geçerli iterasyonda bulunan en iyi turun uzunluğudur.

Küresel feromon güncellemesi, geçerli iterasyondaki en iyi sonuca sahip karıncanın izlediği yolun feromon düzeyinin arttırılmasından oluşur ve iterasyonlarda bulunan en iyi sonuçların belli bir oranda ileriki iterasyonlara aktarılmasını sağlar. Küresel bir feromon güncelleme kuralı iki aşamada uygulanır:

İlk aşama: Feromon izinin otomatik olarak azaldığı bir buharlaşma fazı. Her bir feromon değeri sabit bir oranda azaltılır.

$$\tau_{ij} = (1-p)\tau_{ij},\ \forall i,j\ \in [1,n] \tag{2.7}$$

Feromon azalma oranı $p \in [0,1]$ ile temsil edilir. Buharlaşmanın amacı, tüm karıncalar için başlangıçta bulunan iyi çözümlere doğru erken bir yakınsamadan kaçınmak ve daha sonra

arama alanındaki çeşitlendirmeyi destekleyerek olası yeni güzergahların keşfinin yapılabilmesini sağlamaktır.

İkinci aşama: Feromon izinin üretilen çözümlere göre güncellendiği bir pekiştirme aşaması. Feromon güncelleme işlemi, çözüm arama süreci devam ederken adım adım ya da sadece bir karınca çözüme ulaşınca gecikmeli şekilde online yapılabilir. Bu aşamada uygulanan en popüler yaklaşım ise bütün karıncalar bir çözüm ürettiğinde yapılan offline feromon güncellemesidir. Bu yaklaşımda farklı stratejiler kullanılabilir:

- Kaliteye dayalı feromon güncellemesi: Bu strateji tüm karıncalar arasında bulunan en iyi veya karınca sayısından daha küçük olmak üzere en iyi k adet çözümle ilişkili feromon değerini günceller. Eklenen değerler seçilen çözümlerin kalitesine bağlıdır. Örneğin, en iyi çözüme (π^*) ait her bileşene, bir pozitif Δ değeri eklenir.

$$\tau_{i\pi^*(i)} = \tau_{i\pi^*(i)} + \Delta,\ \forall i\ \in\ [1, \mathrm{n}] \tag{2.8}$$

- Sıra tabanlı feromon güncelleme: Miktarı çözümün sırasına bağlı olarak en iyi k adet çözümün feromon güncellemesi yapabilmesine imkân verilir.
- En kötü feromon güncellemesi: En kötü çözümü üreten karınca çözüm bileşenleri ile ilgili feromon izlerini azaltır.
- Elitist feromon güncellemesi: Şimdiye kadar bulunan en iyi çözüm, aramanın yoğunlaşmasını sağlamak için feromonu günceller.

Amaçlı sezgisel yöntemlerdeki klasik ve yaygın arama bileşenlerine ek olarak bir KKO'nun tasarlanmasındaki asıl konu, aşağıdakilerin belirlenmesidir:

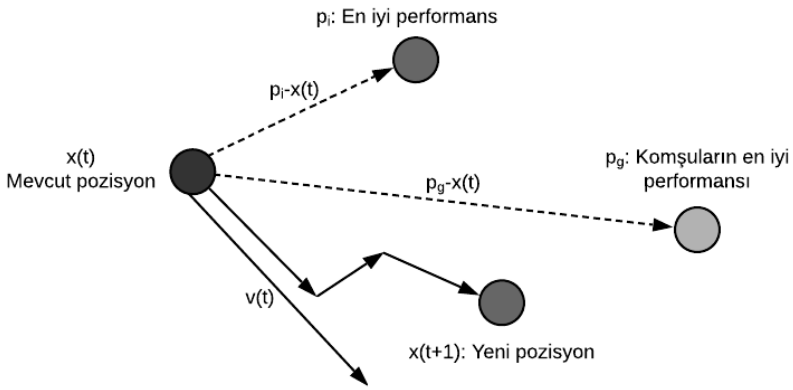
Feromon bilgisi: Feromon modeli, KKO algoritmalarının merkezi bileşenini temsil eder. Feromon izi parametreleri olarak adlandırılan τ, model parametrelerinin bir vektörünün tanımlanmasından oluşur. Feromon değerleri $\tau_i \in \tau$, verilen bir problem için çözümün yapımında ilgili bilgiyi yansıtmalıdır. Genellikle bir çözümün bileşenleri ile ilişkilendirilirler.

Çözüm inşası: Çözüm inşasında ana soru, feromonun yanı sıra araştırmaya rehberlik etmek için kullanılacak yerel sezgisel tanımla ilgilidir. KKO metaforunun nispeten etkin açgözlü algoritmaların mevcut olduğu problemlerin çözümünde uyarlanması kolaydır.

Feromon güncellemesi: Temel olarak feromon bilgisi için takviye öğrenme stratejisi tanımlanmalıdır.

2.1.2. Parçacık Sürü Optimizasyonu

Parçacık sürü optimizasyonu, sürü zekasından ilham alan bir başka rastlantısal popülasyon tabanlı meta-sezgisel optimizasyondur. Yeterince yiyeceğe sahip bir yer bulmak için kuş ve balık gibi doğal organizmaların gerçekleştirdiği davranışları taklit eder. Bu sürülerde, yerel hareketleri kullanan koordineli davranışlar, herhangi bir merkezi kontrol olmadan ortaya çıkmaktadır. PSO sürekli optimizasyon problemlerini çözmek için başarıyla tasarlanmıştır.



Şekil 2.2. Parçacığın hareketi ve hız güncellemesi

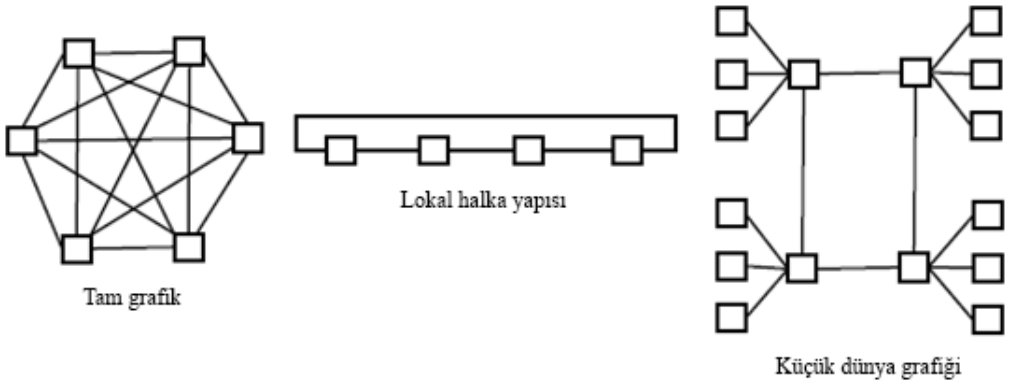
Temel modelde, bir küme D boyutlu bir arama uzayında etrafta uçuşan N adet parçacıktan oluşur. Her parçacık *i*, soruna aday bir çözümdür ve karar alanında x_i vektörüyle temsil edilir. Bir parçacığın uçuş yönü ve adımı anlamına gelen kendi konumu ve hızı vardır. Optimizasyon, parçacıklar arasındaki iş birliğinden faydalanır. Bazı parçacıkların başarısı diğerlerinin davranışlarını etkileyecektir. Her bir parçacık, şu iki faktöre göre, x_i pozisyonunu global optimuma doğru yaklaştırır; $p_i = (p_{i1}, p_{i2}, ..., p_{iD})$ olarak belirtilen kendisinin en iyi pozisyonu ($pbest_i$) ve $p_g = (p_{g1}, p_{g2}, ..., p_{gD})$ olarak gösterilen tüm sürü tarafından ziyaret edilen en iyi pozisyon (gbest) (veya lbest, sürünün belirli bir alt kümesi için en iyi konum). Bir vektör (p_g - x_i), *i* parçacığının mevcut pozisyonu ile komşularının en iyi pozisyonu arasındaki farkı temsil eder.

2.1.2.1. Parçacık komşuluğu

Her parçacık için bir komşuluk ilişkisi tanımlanmalıdır. Bu komşuluk parçacıklar arasındaki sosyal etkiyi ifade eder. Böyle bir komşuluğu tanımlamanın birçok yolu vardır. Geleneksel olarak iki yöntem kullanılır:

Global En İyi Yöntem (gbest): Küresel en iyi yöntemde, komşular bütün parçacık popülasyonu olarak tanımlanır.

Lokal En İyi Yöntem (lbest): Yerel en iyi yöntemde, belirli bir topoloji sürüyle ilişkilendirilir. Bu nedenle, bir parçacığın komşuluğu doğrudan bağlı parçacıkların kümesidir. Parçacıklar izole edildiğinde komşuluk durumu oluşmayabilir. Şekil 2.3, üç farklı topolojiyi göstermektedir: tam grafik, halka grafiği ve küçük dünya grafiği. Popülasyonun tamamının komşuluk oluşturması ile tam, komşuluk yapısını tanımlayan her parçacığın iki komşuya sahip olduğu halka yapısı ve bir orta düzey küçük dünya grafiği topolojisi bulunmaktadır. Bu model, istikrarlı bir konfigürasyonun homojen alt popülasyonlardan oluşacağı, nüfus üyelerinin birbirlerini karşılıklı taklitlerine dayanan sosyal bilim modellerine benzemektedir. Her alt popülasyon, hemfikir bir kültür oluşturacak bir sosyometrik bölge tanımlayacaktır. Aynı sosyometrik bölgeye giren bireyler benzer ve farklı bölgelere ait bireyler farklı olma eğilimindedir.



Şekil 2.3. Parçacık komşuluğu topolojileri (Talbi, 2009)

Kullanılan komşuluğa göre, bir lider (yani gbest veya lbest), bir parçacığın aranmasına karar alanın daha iyi bölgelerine doğru yol göstermek için kullanılan parçacığı temsil eder. Bir parçacık üç vektör içerir:

• *x*-vektörü, parçacığın mevcut pozisyonunu (konumunu) arama alanına kaydeder.

• *p*-vektörü, parçacık tarafından belirli bir zamana kadar bulunan en iyi çözümün konumunu kaydeder.

• *v*-vektörü, rahatsız edildiğinde partikülün hareket edeceği bir gradyan (yön) içerir.

• İki uygunluk değeri: *x*-fitness, *x*-vektörünün uygunluğunu ve *p*-fitness, *p*-vektörünün uygunluğunu kaydeder.

Bir parçacık sürüsü, bireysel hücrelerin (PSO'daki parçacıklar) güncellemelerinin paralel olarak yapıldığı hücresel bir otomat olarak görülebilir. Her yeni hücre değeri yalnızca hücrenin ve komşularının eski değerine bağlıdır ve tüm hücreler aynı kurallar kullanılarak güncellenir. Her yinelemede, her bir parçacık aşağıdaki işlemleri uygulayacaktır:

Hız güncellemesi: Parçacıklara uygulanacak değişimin miktarını tanımlayan hız; ρ_1 ve ρ_2'nin [0, 1] aralığında iki rastgele değişken olacağı şekilde tanımlanır.

$$v_i(t) = v_i(t-1) + \rho_1 C_1 \times \left(p_i - x_i(t-1)\right) + \rho_2 C_2 \times (p_g - x_i(t-1)) \qquad (2.9)$$

C_1 ve C_2 sabitleri öğrenme faktörlerini temsil eder. Bir parçacığın kendi başarısına yönelik veya komşularının başarısına yönelik çekiciliğini temsil ederler. C_1 parametresi, bir parçacığın kendi başarısına yönelik çekiciliğini temsil eden bilişsel öğrenme faktörüdür. C_2 parametresi ise, bir partikülün komşularının başarısına yönelik çekiciliğini temsil eden sosyal öğrenme faktörüdür. Hız, partikülün gitmesi gereken yönü ve mesafeyi tanımlar (Bkz. Şekil 2.2). Bu formül, bireyin sosyal-psikolojik eğiliminin diğer bireylerin başarılarına öykündüğü, insan sosyalliğinin temel bir yönünü yansıtmaktadır. Hız güncelleme formülünün ardından, bir parçacık, p_i ve p_g'nin ağırlıklı ortalaması olarak tanımlanan nokta etrafında dönecektir.

$$\frac{\rho_1 p_i + \rho_2 p_g}{\rho_1 + \rho_2} \qquad (2.10)$$

v_i'nin elemanları, sistemin rastlantısallığı nedeniyle patlamamaları için maksimum bir değer $[-V_{max}, +V_{max}]$ ile sınırlıdır. Eğer v_i hızı V_{max} maksimum hız değerini aşarsa tekrar V_{max} durumuna getirilir.

Genellikle hız güncelleme prosedüründe, bir eylemsizlik ağırlığı *w* önceki hıza eklenir:

$$v_i(t) = w \times v_i(t-1) + \rho_1 \times \left(p_i - x_i(t-1)\right) + \rho_2 \times (p_g - x_i(t-1)) \quad (2.11)$$

Eylemsizlik ağırlığı *w*, önceki hızın mevcut hız üzerindeki etkisini kontrol edecektir. Eylemsizlik ağırlığının büyük değerleri için, önceki hızların etkisi çok daha yüksek olacaktır. Bu nedenle, eylemsizlik ağırlığı, küresel keşif ve yerel kullanım arasında bir takası temsil eder. Büyük bir eylemsizlik ağırlığı, tüm arama alanındaki aramaları çeşitlendirerek küresel araştırmayı teşvik ederken, daha küçük bir eylemsizlik ağırlık değeri ise mevcut bölgedeki aramayı yoğunlaştırarak yerel kullanımı teşvik eder.

Pozisyon güncellemesi: Arama uzayında her parçacık kendi koordinatlarını günceller. Sonra yeni pozisyonuna doğru hareket eder.

$$x_i(t) = x_i(t-1) + v_i(t) \quad (2.12)$$

En iyi bulunan parçacıkların güncellenmesi: Her parçacık, potansiyel olarak en iyi yerel çözümü güncelleyecektir:

$$\text{Eğer } f(x_i) < \text{pbest}_i \text{ ise, } p_i = x_i \quad (2.13)$$

Ayrıca, sürünün en iyi küresel çözümü güncellenir:

$$\text{Eğer } f(x_i) < \text{gbest}_i \text{ ise, } g_i = x_i \quad (2.14)$$

Dolayısıyla, her bir yinelemede, her bir parçacık kendi deneyimine ve komşu olduğu parçacıkların konumuna göre pozisyonunu değiştirecektir.

Herhangi bir sürü istihbarat konseptine gelince, ajanlar (PSO için parçacıklar) yapılan arama ile ilgili deneyimleri paylaşmak için bilgi alışverişinde bulunuyorlar. Tüm sistemin davranışı, bu basit ajanların etkileşiminden kaynaklanmaktadır. PSO'da, paylaşılan bilgi en iyi küresel çözümden (gbest) oluşur.

Çizelge 2.1. PSO algoritması parametreleri

Parametre	**Rolü**	**Değerleri**
n	Parçacık sayısı	[20,60]
τ_1, τ_2	Hızlanma sabitleri	≤ 2.0
k	Komşuluk boyutu	[2, n×(n-1)/2]
w	Atalet ağırlığı	[0.4,0.9]

Çizelge 2.1'deki parametreleri kullanan Parçacık Sürü Optimizasyon algoritmasına ait şema aşağıdaki gibidir:

Tüm sürünün rastgele başlatılması;
Repeat
    Değerlendirme
    For (bütün parçacıklar)
        Hız güncellemesi
        Yeni pozisyonlara hareket
    If $f(x_i) < \text{pbest}_i$ ise, $p_i = x_i$
    If $f(x_i) < \text{gbest}_i$ ise, $g_i = x_i$
    Güncelle
    End

Until Durma kriteri

2.1.3. Benzetilmiş Tavlama

Bilgisayar bilimlerinde, özellikle optimizasyon alanında kullanılan algoritmalardan birisidir. Demir madeni işlenirken uygulanan işlemlerden biri olan, demiri ısıtıp ardından soğutmaya bırakmak anlamına gelen demir tavlama işleminden esinlenmiştir. Algoritmanın amacı, herhangi bir problem için genel iyileştirme elde etmektir. Başka bir deyişle, herhangi bir fonksiyonun ya da ölçümün genel minimum veya maksimum değerini elde etmeyi amaçlamaktadır.

Benzetilmiş tavlama, tavlama işleminde bir maddenin, güçlü kristal bir yapı elde etmek için ısıtılması ve yavaş yavaş soğutulmasını gerektiren istatistiksel mekanik prensiplerine dayanmaktadır. Yapının gücü, soğutma metallerinin oranına bağlıdır. İlk sıcaklık yeterince yüksek değilse veya hızlı bir soğutma uygulanırsa, kusurlar ortaya çıkar. Bu durumda, soğuyan katı her sıcaklıkta termal dengeye ulaşmayacaktır. Güçlü kristaller, dikkatli ve yavaş soğutma yapılarak üretilir. BT algoritması, soğutma işlemine maruz kalan bir sistemdeki enerji değişimlerini denge durumuna (sabit donmuş duruma) yaklaşıncaya kadar simüle eder.

Çizelge 2.2, fiziksel sistem ile optimizasyon problemi arasındaki benzeşimi göstermektedir. Sorunun nesnel fonksiyonu, sistemin enerji durumuna benzer. Optimizasyon probleminin çözümü sistemin durumuna karşılık gelir. Bir problemin çözümü ile ilgili karar değişkenleri moleküler pozisyonlarla benzer şekildedir. Global optimum, sistemin temel haline karşılık gelir. Yerel bir minimum bulmak, yarı kararlı bir hale ulaşıldığını gösterir.

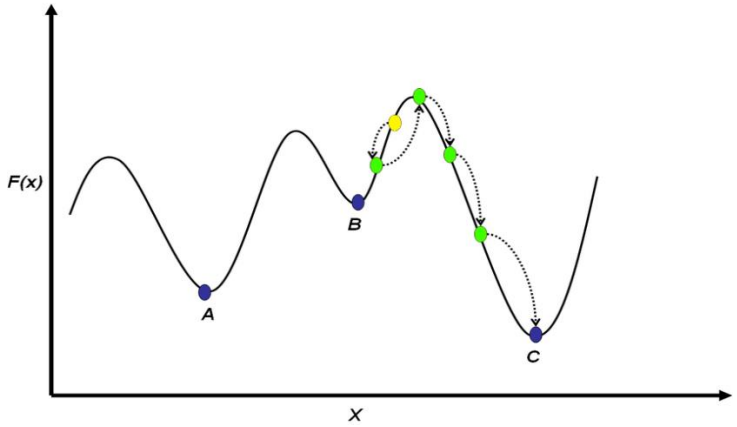
Çizelge 2.2. Fiziksel sistem ile optimizasyon problemi arasındaki benzeşim

Fiziksel Sistem	*Optimizasyon Problemi*
Sistem durumu	Çözüm
Moleküler pozisyonlar	Karar değişkenleri
Enerji	Amaç fonksiyonu
Temel hal	Global optimal çözüm
Yarı kararlı hal	Lokal optimum
Seri söndürme	Lokal arama
Sıcaklık	Kontrol parametresi
Dikkatli tavlama	Benzetilmiş tavlama

Benzetilmiş tavlama, bazı koşullar altında bir çözümün bozulmasını sağlayan olasılıksal bir algoritmadır. Amaç, yerel en iyi durumdan kaçmak ve böylece yakınsamayı geciktirmektir. BT, algoritmanın arama sırasında toplanan herhangi bir bilgiyi kullanmadığı, hafıza bilgisi içermeyen bir algoritmadır. Başlangıç çözümünden itibaren bu algoritma, farklı iterasyonlarla ilerler. Her iterasyonda, bir rastgele komşu yaratılır. Maliyet fonksiyonunu geliştiren hareketler her zaman kabul edilir. Aksi takdirde, komşu mevcut sıcaklığa ve amaç fonksiyonunun bozulma miktarına ΔE bağlı olarak verilen bir olasılıkla seçilir. ΔE, mevcut çözüm ile oluşturulan komşu çözüm arasındaki objektif değerdeki enerji farkını temsil eder. Algoritma ilerledikçe, bu tür hareketlerin kabul edilme olasılığı azalır (Şekil 2.4). Sıcaklık ne kadar yüksek olursa, en kötü hareketi kabul etme olasılığı o kadar önemli olur. Belirli bir sıcaklıkta, amaç fonksiyonun artışı ne kadar düşük olursa, hareketi kabul etme olasılığı o kadar önemli olur. Daha iyi bir hamle her zaman kabul edilir. Genel olarak bu olasılık Boltzmann dağılımını izler:

$$P(\Delta E, T) = e^{-\frac{f(s')-f(s)}{T}} \tag{2.15}$$

İyileştirici olmayan çözümleri kabul etme olasılığını belirlemek için sıcaklık adında bir kontrol parametresi kullanır. Belirli bir sıcaklık seviyesinde, birçok deneme yapılır.



Şekil 2.4. Benzetilmiş tavlama yerel çözümlerden kaçınmaktadır (Hosny, 2012)

Bir denge durumuna ulaşıldığında, sıcaklık programlı bir şekilde soğutulur. Sıcaklık kademeli olarak azaltıldıktan sonra arama bittiğinde, az sayıda iyileştirici olmayan çözüm kabul edilir. Aşağıdaki algoritma, BT algoritmasının şablonunu göstermektedir.

Input: Soğutma programı
Başlangıç çözümünün yaratılması
Başlangıç sıcaklığı
Repeat
    Repeat Belirli sıcaklıkta
    Rastgele komşu oluşturma
    $\Delta E = f(s') - f(s)$
    If $\Delta E \leq 0$ **Then** $s = s'$ Komşu çözümü kabul et
    Else s' *nün* $e^{-\frac{\Delta E}{T}}$ olasılığını kabul et
    Until Denge durumu
    Sıcaklık güncellemesi
Until Durma kriteri
Output: Bulunen en iyi çözüm

2.1.3.1. Hareket kabulü

Sistem, ilerleme kaydetmeyen bir komşunun olası kabulü nedeniyle yerel çözümlerden kaçınabilir. Gelişmeyen bir komşuyu kabul etme olasılığı, T sıcaklığıyla doğru ve amaç fonksiyonun ΔE değişmesi ile ters orantılıdır.

Termodinamiğin yasası gereği T sıcaklığında enerji miktarındaki artış; $P(\Delta \mathrm{E}, \mathrm{T}) = \exp(-\Delta E/kt)$ olarak belirtilir. Boltzmann sabiti olarak bilinen k değerini kullanır. Böylelikle, ilerleme kaydetmeyen hareketlerin kabul olasılığı:

$$P(\Delta \mathrm{E}, \mathrm{T}) = \exp\left(-\frac{\Delta E}{kt}\right) > R \quad (2.16)$$

ΔE, ölçüm fonksiyonundaki değişim miktarı, T mevcut sıcaklık ve R, 0 ile 1 arasında bir rasgele sayıdır.

Yüksek sıcaklıklarda, daha kötü hareketleri kabul etme olasılığı yüksektir. $T = \infty$ ise, tüm hareketler kabul edilir. Bu durum bir arazide rastgele dolaşmak gibidir. Eğer $T = 0$ ise, daha kötü hamleler kabul edilmez ve arama, yerel aramayla eşdeğerdir. Ayrıca, çözüm kalitesinde büyük bir bozulmanın kabul edilme olasılığı, Boltzmann dağılımına göre 0'a doğru üssel olarak azalır.

2.1.3.2. Programlı soğutma

Soğutma programı algoritmanın her adımındaki sıcaklık miktarını belirler. Benzetilmiş tavlama algoritmasının başarısında önemli rol oynar. Algoritmanın performansı büyük oranda soğutma tercihine bağlıdır. Bir soğutma programı tanımlarken göz önünde bulundurulacak parametreler; başlangıç sıcaklığı, denge durumu, bir soğutma fonksiyonu ve durma kriterlerini tanımlayan son sıcaklıktır.

❖ Başlangıç Sıcaklığı

Başlangıç durumundaki sıcaklık, aramanın biçimini etkilemektedir. Başlangıç sıcaklığı, belirli bir sürede rastgele bir arama yapmaya sebebiyet verecek kadar yüksek olmamalı, ancak hamlelerin neredeyse komşuluk durumuna taşınmalarını sağlayacak kadar yüksek olmalıdır. Bu parametreyi ele almak için kullanılabilecek üç ana strateji vardır:

- Tümünü kabul etme: İlk durumdaki sıcaklık, algoritmanın başlangıç aşaması sırasında tüm komşuları kabul edecek kadar yüksek olarak ayarlanır. Bu stratejinin ana dezavantajı yüksek işlemsel maliyettir.
- Kabul sapması: Sıcaklık, $k\sigma$ denklemi kullanılarak ön hazırlık ile hesaplanır. σ, fonksiyonların değerleri arasındaki farkın standart sapmasını ve $k = -3/\ln(p)$ ise 3σ değerinden büyük olan p'nin kabul olasılığını temsil eder.

- Kabul oranı: Başlangıç sıcaklığı, çözümlerin kabul oranının önceden belirlenmiş bir a_0 değerinden daha büyük olacağı şekilde tanımlanmıştır. Örneğin, başlangıç sıcaklığı, kabul oranının [%40, %50] aralığında olduğu şekilde başlatılmalıdır.

$$T_0 = \frac{\Delta^+}{\ln\left(m_1(a_0-1)/m_2+a_0\right)} \tag{2.17}$$

Sırasıyla m_1 ve m_2, ön deneylerde azaltılması ve arttırılması gereken çözüm sayıları ve Δ^+ artış gösteren amaç fonksiyonlarının ortalamasıdır.

❖ Denge Durumu

Bütün sıcaklıklarda bir denge durumuna ulaşmak için, yeterli sayıda geçiş uygulanmalıdır. Teori, her sıcaklıktaki iterasyon sayısının, pratikte uygulanması zor bir strateji olan problem boyutuna katlanabileceğini göstermektedir. Yinelemelerin sayısı, örnek problemin boyutuna göre ve özellikle komşuluk boyutuyla orantılı olarak ayarlanmalıdır. Ziyaret edilen geçişlerin sayısı aşağıdaki gibi olabilir:

Statik: Statik bir stratejide, geçiş sayısı arama başlamadan önce belirlenir. Örneğin, komşuluğun belirli bir kısmı keşfedilmiştir. Bu nedenle, bir çözümden üretilen komşuların sayısı belirlenen kısım ile komşuluk boyutunun çarpımıdır. Komşuluğun oranı ne kadar kayda değer ise, hesaplama maliyeti de o kadar yüksek ve elde edilen sonuçlar da o kadar iyi olur.

Adaptif: Üretilen komşuların sayısı aramanın özelliklerine bağlı olacaktır. Örneğin, her sıcaklıkta denge durumuna ulaşmak gerekli değildir. Dengenin sağlanmadığı benzetilmiş tavlama algoritmaları kullanılabilir. Soğutma programı, iyileştirici bir komşu çözüm üretilir üretilmez uygulanabilir. Bu özellik, elde edilen çözümlerin kalitesinden ödün vermeden hesaplama süresinin kısalmasına neden olabilir.

❖ Soğutma

Benzetilmiş tavlama algoritmasında sıcaklık kademeli olarak düşürülür.

$$T_i = 0, \forall_i \quad ve \quad \lim_{i\to\infty} T_i = 0 \tag{2.18}$$

Elde edilen çözümlerin kalitesi ile soğutma programının hızı arasında daima bir uzlaşım vardır. Sıcaklık yavaşça düşürülürse daha iyi çözümler elde edilir, ancak daha önemli bir hesaplama süresi olur. Sıcaklık farklı şekillerde güncellenebilir:

- Doğrusal: Doğrusal soğutma sürecinde *T* sıcaklığı aşağıdaki gibi güncellenir:

$$T_i = T_0 - i \times \beta \tag{2.19}$$

T_i, bir i iterasonundaki sıcaklığı, β ise belirtilen sabit bir değeri temsil eder.

- Geometrik: Sıcaklık aşağıdaki formüle göre güncellenir:

$$T = \alpha T \tag{2.20}$$

Bu en popüler soğutma fonksiyonudur. Deneyler α değerinin 0,5 ila 0,99 arasında olması gerektiğini göstermiştir.

- Logaritmik: Bu program pratikte uygulanamayacak kadar yavaştır ancak yakınsamayı bir global optimum haline getirme özelliğine sahiptir.

$$T_i = \frac{T_0}{\log(i)} \tag{2.21}$$

- Çok yavaş azaltma: Bir soğutma programındaki ana ödünleşim, birkaç sıcaklıkta çok sayıda yinelemenin veya birçok sıcaklıkta az sayıda yinelemenin kullanılmasıdır. Bu çok yavaş azalan fonksiyonda, her sıcaklıkta sadece bir iterasyona izin verilir.

$$T_{i+1} = \frac{T_0}{1+\beta T_i} \tag{2.22}$$

- Monoton olmayan: Tipik soğutma programları, monoton sıcaklıkları kullanır. Bazı sıcaklığın tekrar arttırıldığı monoton olmayan programlama şemaları önerilebilir. Bu, arama alanındaki çeşitliliği teşvik edecektir. Bazı arama alanları için en uygun zamanlama monoton olmayandır.

- Adaptif: Soğutma programlarının çoğu, soğutma programının tamamen önceden tanımlanmış olması açısından statiktir. Bu durumda, soğutma programı arama sahasının özelliklerine karşı kördür. Adaptif bir soğutma programında, düşme hızı dinamiktir ve arama sırasında elde edilen bazı bilgilere dayanır. Yüksek sıcaklıklarda az sayıda ve

düşük sıcaklıklarda çok sayıda iterasyonun gerçekleştirildiği bir dinamik soğutma programı kullanılabilir.

❖ Durma Kriteri

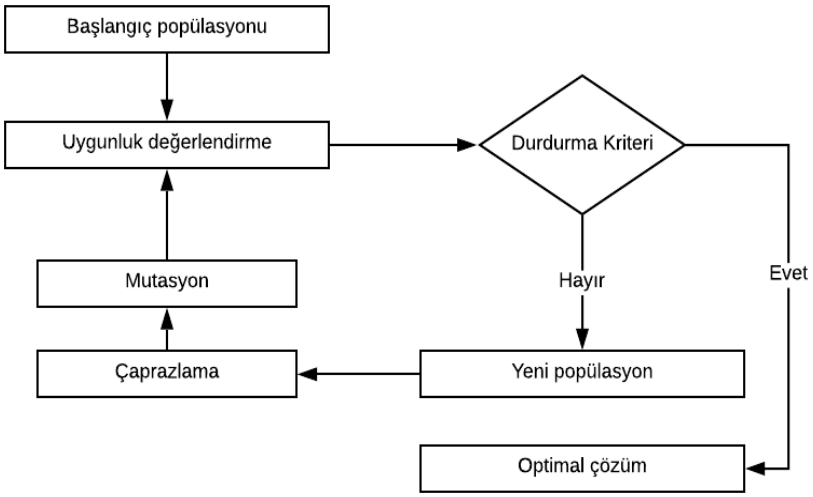
Durma koşuluyla ilgili olarak teori, 0'a eşit bir son sıcaklık önerir. Uygulamada, bir hareketi kabul etme olasılığı göz ardı edilebildiğinde, arama durdurulabilir. Farklı durma kriterleri uygulanabilmektedir. Son sıcaklığa erişme, en popüler durma kriterleridir. Bu sıcaklık sıfıra yakın derecede düşük olmalıdır. Bulunan en iyi çözümün iyileştirilmesi olmadan önceden belirlenmiş sayıda tekrarlamanın elde edilmesi de durma kriteri olarak kullanılabilir. Her sıcaklık kabul edildiğinde önceden belirlenmiş sayıda komşuluk yüzdesine erişme durumunda ise, eğer sayaç belirlenmiş limite gelirse benzetilmiş tavlama algoritması durur.

2.1.4. Genetik Algoritma

Genetik algoritmalar, doğal sistemlerin uyarlanabilir süreçlerini anlamak için J. Holland tarafından geliştirilmiştir. Başlangıçta, kromozomlardan oluşan popülasyonları bazı operatörler kullanarak yeni popülasyonlara dönüştürmek için tasarlanmıştır. Daha sonra birtakım değişikliklere uğrayarak optimizasyon ve makine öğrenmesine uygulanmıştır.

Genetik algoritmalar çok popüler bir evrimsel algoritma sınıfıdır. Bir GA genellikle, önemli bir rol oynayan iki çözüme bir çaprazlama operatörünü ve çeşitliliği artırmak için bireysel içeriği rastgele değiştiren bir mutasyon operatörünü uygular. GA'lar aslında orantılı seçim olan olasılıklı bir seçilim kullanır. Seçimi belirleyen yer değiştirme nesilseldir, yani ebeveynler sistematik olarak yavrular tarafından değiştirilir. Çaprazlama operatörü, mutasyon bit değiştirirken n-noktalı veya tek biçimli çaprazlamayı temel alır. Mutasyon operatörüne sabit bir olasılık uygulanır.

Evrimsel bir algoritma tasarlamanın ana arama bileşenleri şunlardır; genlerin sunumu, popülasyon başlangıcı, amaç fonksiyonu, seleksiyon, mutasyon ve çaprazlama ile yeniden üretim, nesillerin yer değiştirmesi ve durma kriteri (Şekil 2.5).



Şekil 2.5. Genetik algoritma akış diyagramı

2.1.4.1. Seçim metotları

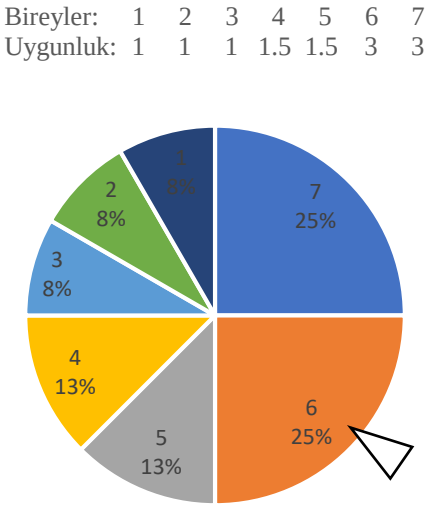
Seçim mekanizması genetik algoritmalarda ana arama bileşenlerinden biridir. Birey ne kadar iyiyse, ebeveyn olma şansı da o kadar yüksektir prensibine göre seçim yapılmaktadır. Bununla birlikte, en kötü bireyler atılmamalı ve seçilme şansları bulunmalıdır. Bu yararlı genetik materyallerin oluşmasına yol açabilir.

Seçim stratejisi, hangi bireylerin üreme için seçildiğini ve seçilen her bireyin kaç yavru ürettiğini belirler. Bireylere uygunluk ataması iki farklı yol ile yapılabilir. Bu yollar, mutlak uygunluğun bireylerle ilişkilendirildiği orantılı uygunluk ataması ve bağıl uygunlukların bireylerle ilişkilendirildiği sıra tabanlı bir uygunluk atama işlemidir. Örneğin, popülasyondaki bir sıralama, bireylerin azalan bir sıralamadaki sırasına göre her birey ile ilişkilendirilir. Daha sonra, ebeveynler uygunluklarına göre seçilir.

- Rulet Çarkı Seçimi:

En sık kullanılan seçim stratejisidir. Her bireye, bağıl uygunluk ile orantılı bir seçim olasılığı atayacaktır. Her bir bireyin grafiğe uygunluğuyla orantılı olarak bir alan tahsis edildiği bir pasta grafiği varsayalım (Şekil 2.6). Pasta etrafına bir dış rulet tekerleği yerleştirilir.

Bireylerin seçimi, rulet çarkının bağımsız dönüşleri ile yapılır. Her tur tek bir birey seçecektir. Daha iyi bireylerin daha fazla alana sahip olmaları ve daha sonra seçilme şansları daha fazladır.



Şekil 2.6. Rulet çarkı seçimi pasta grafiği

Rulet çarkı seçiminde seçilen kişiler, araştırmanın başlangıcında erken bir yakınlaşmaya ve çeşitlilik kaybına neden olabilecek bir eğilim göstereceklerdir. Ayrıca, tüm bireyler eşit derecede uygun olduğunda, bu seçim stratejisi en iyi bireyleri seçmek için yeterli olmayabilir.

- Stokastik Evrensel Örnekleme:

Rulet seçim stratejisindeki sapmayı azaltmak için stokastik evrensel örnekleme kullanılabilir. Eşit aralıklı işaretçilerle birlikte pastanın etrafına bir dış rulet tekerleği yerleştirilir. Bu stratejide, rulet tekerleğinin tek bir dönüşü aynı anda tüm üreme bireylerini seçecektir.

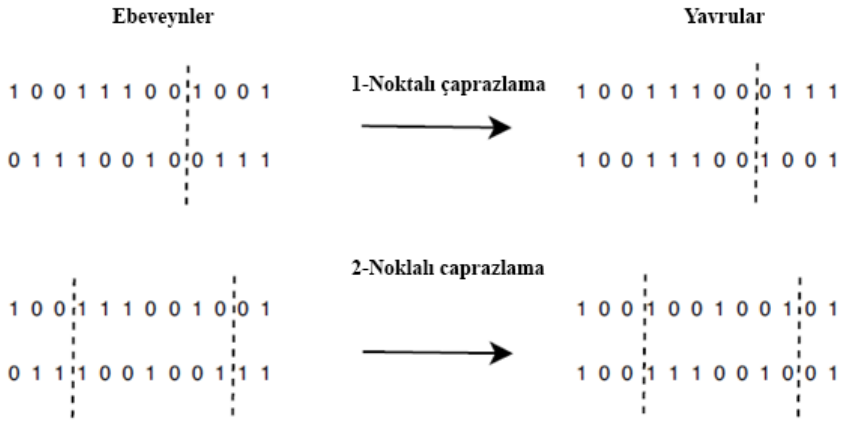
- Sıra Tabanlı Seçim:

Bir bireyin uygunluk değerini kullanmak yerine, bireyin sırası kullanılır. Fonksiyon, yüksek rütbeli bireylere yönelimlidir. Popülasyonda sıralama yapılarak üst sıradaki bireyler seçilir.

2.1.4.2. Çaprazlama

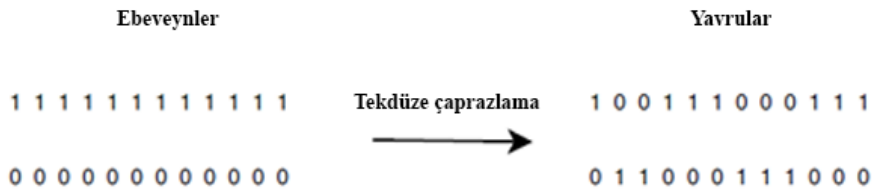
Çaprazlama operatörlerinin rolü, yavruları oluşturmak için iki ebeveynin bazı özelliklerini miras almaktır. Bir karakteri belirleyen anneden ve babadan alınan gen iki tanedir. Bir yavrunun özellikleri bu genlerin çaprazlanması sonucu oluşan genotipin aktarılmasıyla oluşur. Çaprazlama operatörünün temel özelliği kalıtımdır. Çaprazlama operatörü, her iki ebeveynden de genetik bir materyal almalıdır. Bu materyaller ile geçerli çözümler üretmektedir.

Çaprazlama oranı p_c, bir çaprazlama operatörünün ebeveynlerden alacağı gen oranını temsil eder. Bu oran için en iyi parametre değeri, popülasyon büyüklüğü, mutasyon olasılığı ve seçim prosedürü gibi aralarındaki diğer parametrelerle ilgilidir. En sık kullanılan oranlar [0,45, 0,95] aralığındadır. Çaprazlama oranı için uyarlanabilir teknikler de faydalı olabilir.



Şekil 2.7. Tek ve n noktalı çaprazlama operatörleri

Temel çaprazlama operatörü, 1-noktalı ve genel olarak, n-noktalı çaprazlamadır. Bu operatörler (Şekil 2.7) başlangıçta binary gösterimler için önerilmişlerdir. Tek noktalı çaprazlamada, bir kesim yeri rastgele seçilir ve ebeveynlerin bölümleri değiştirilerek iki yavru oluşturulur. Bu kesim noktaları çoğaltılarak farklı gen aktarımları yapabilmek mümkündür.



Şekil 2.8. Tekdüze çaprazlama operatörü

Tekdüze (üniform) çaprazlama kullanılarak, bölümlerin boyutuna bakılmaksızın iki kişi yeniden birleştirilebilir. Yavruların her bir elemanı, her iki ebeveynden rastgele seçilmiştir. (Şekil 2.8) Her bir ebeveyn, yavruların üretilmesine eşit katkıda bulunacaktır.

Bu çaprazlana yöntemlerine ek olarak gerçek değerli hesaplamalarda ortalama ve ebeveyn merkezli çaprazlamalar da yapılabilir. Ortalama merkezli çaprazlamada, bireyler ebeveynlerinin merkezlerine daha yakın üretilir. Ebeveyn merkezli yeniden birleşimde yavrular ebeveynlerine daha yakın üretilir. Her ebeveyne, komşuluğunda yavru oluşturmak için eşit bir olasılık verilir.

2.1.4.3. Mutasyon

Mutasyon tek bireye etki eden bir operatördür. Mutasyonlar, popülasyondaki seçilmiş kişilerdeki küçük değişikliklerini temsil eder. p_m olasılığı, gösterimin her bir elemanını (gen) mutasyona uğratma olasılığını tanımlar. Aynı zamanda sadece bir geni de etkileyebilir. Genel olarak, bu olasılık için küçük değerler önerilmektedir. ($p_m \in [0.001, 0.01]$) Bazı stratejiler mutasyon olasılığını $1 / k$ olarak başlatır, burada k karar değişkenlerinin sayısıdır, yani ortalama olarak sadece bir değişken mutasyona uğrar.

Bir mutasyon operatörünün tasarımında veya kullanımında dikkate alınması gereken bazı önemli noktalar vardır. Mutasyon operatörü, arama alanının her çözümüne ulaşılmasına izin vermelidir. Ayrıca bu operatör geçerli çözümler üretmelidir. Sınırlı optimizasyon problemleri için bu her zaman mümkün olmamaktadır. Mutasyon minimal bir değişiklik yapmalıdır. Mutasyonun boyutu önemlidir ve kontrol edilebilir olmalıdır.

Meta-sezgisel yöntemlerde komşuluk tanımına uygun şekilde, bir mutasyon operatörünü nitelemek zorunda olan temel özellik yerelliktir. Yerellik, genotip denen gösterimde değişiklik yapıldığında bunun çözüm (fenotip) üzerindeki etkisidir. Genotipte küçük değişiklikler yapıldığında, fenotip küçük değişiklikler göstermelidir. Bu durumda, mutasyonun güçlü bir yerelliğe sahip olduğu söylenir. Bu sayede, evrimsel bir algoritma problem alanında anlamlı bir araştırma yapacaktır. Zayıf yerellik durumunda, arama süreci alandaki rasgele bir aramaya dönüşecektir.

Evrimsel algoritmalardaki mutasyon meta-sezgisel komşuluk operatörleri ile ilgilidir. Bu nedenle, komşuluk yapısı tanımları mutasyon operatörleri olarak yeniden kullanılabilir. İkili sistemdeki mutasyonda, yaygın olarak kullanılan mutasyon 0'dan 1'e veya 1'den 0'a bit çevirme operatörü olarak tanımlanır. Kesikli temsilde, bir elemanla ilişkilendirilen değeri alfabenin başka bir değeri ile değiştirmekten oluşur. Sıra tabanlı gösterimlerdeki mutasyon, genellikle takas, ters çevirme veya ekleme operatörlerine dayanır.

Parçalama ağaçlarının temsil olarak kullanıldığı genetik algoritmalarda, bazı farklı mutasyon biçimleri aşağıdaki gibi tanımlanabilir:

Büyüme: Bir uç düğüm rastgele seçilir ve rastgele oluşturulmuş bir alt ağaç ile değiştirilir.
Daralma: Bir iç düğüm rastgele seçilir ve rastgele oluşturulmuş bir uç düğümle değiştirilir.
Değişme: Bir iç düğüm rastgele seçilir, alt ağaçlarından ikisi rastgele seçilir ve ağaçtaki konumları değiştirilir.
Döngü: İçte veya uçtaki bir ek bir düğüm rastgele seçilir ve aynı sayıda argüman içeren rastgele bir düğümle değiştirilir.

Gerçek değerli vektörler için birçok farklı mutasyon operatörü vardır. En çok kullanılan mutasyon operatörleri sınıfı şu şekildedir:

$$x' = x + M \tag{2.23}$$

M tek tip rastgele, Gaussian dağılımına göre veya polinom mutasyon gibi farklı formlarda olabilen rastgele bir değerdir. Tek tip rastgele mutasyonda, $[a, b]$ aralığında bir rastgele değişken üretilir. Genellikle a parametresi $-b$ değerine eşittir. Bir yavru, b'nin kullanıcı tarafından tanımlanan bir sabit olduğu $x + U(-b, b)^n$ formülü ile bulunur.

Normal dağıtılmış mutasyonda bir Gaussian dağılımı olan $M = N(0, \sigma)$ kullanılır. N, ortalamaları 0 ve standart sapmaları σ olan bir bağımsız rastgele sayıların vektörüdür. En popüler mutasyon şeması budur. Polinom mutasyonda ise, $x_i' = x_i + (x_i^u - x_i^L)\delta_i$ ile yavrular üretilir. δ_i parametresi polinom olasılığı dağılımı ile bulunur.

2.1.4.4. Yer değiştirme

Yer değiştirme aşaması hem ebeveynin hem de yavru popülasyonların hayatta kalan kısmının seçilmesiyle ilgilidir. Nüfusun büyüklüğü sabit olduğundan, bir seçim stratejisine göre bireylerin popülasyonda bulunma durumu belirlenir.

Nesil değiştirme stratejisinde, yer değiştirme tüm nüfusu ilgilendirir. Yavru popülasyon sistematik olarak ana popülasyonun yerini alacaktır. Kararlı durum stratejisinde ise, her bir nesilde yalnızca bir yavru oluşturulur. Bu yavru, ebeveyn nüfusunun en kötü kişisinin yerini alır. Bunların yanında birçok yer değiştirme şeması mevcuttur. Elitist yöntemde her zaman ebeveynlerden ve yavrulardan en iyi bireyler seçilmektedir. Bu yaklaşım daha hızlı bir yakınlaşmaya yol açar ve beklenenden daha erken bir yakınlaşma durumu oluşabilir. Bazen, örnekleme hatası probleminden kaçınmak için kötü bireyleri seçmek gerekir. Bu yenileme stratejileri rastgele ya da önceden belirlenmiş şekilde olabilir.

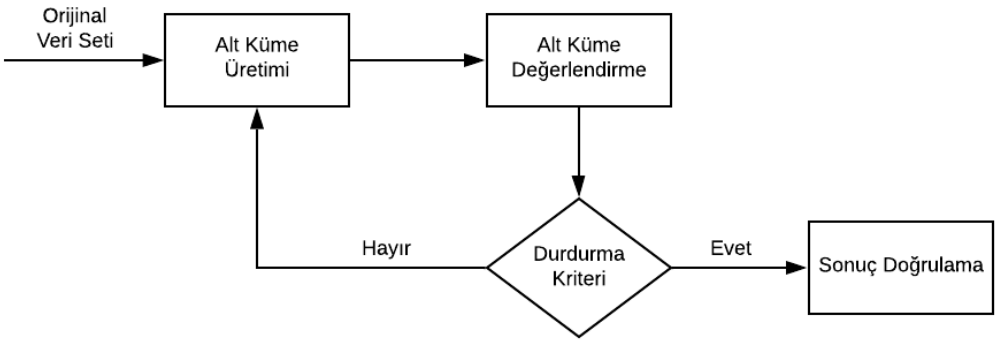
2.2. Özellik Seçimi

Özellik seçimi (feature selection) bir veri setini temsil eden bir alt kümenin belirlenmesi ve bu veriyi en iyi ifade eden değişkenlerin ayrıştırılmasıdır. Bu işlem kullanılan algoritmaya uygun şekilde özellikleri tarayarak *n* adet özellikten en iyi *k* tanesini seçer. Böylelikle özellik sayısı azaltılmış olur ve problem çözümünde çeşitli faydalar sağlar.

Özellik seçimi, nitelik kümesinin boyutunu düşürerek veri analizi için kullanılan algoritmanın daha hızlı çalışmasını sağlar. Gürültülü veya eksik verileri ayrıştırarak verinin kalitesini arttırır. Veri kümesini daha basit hale getirerek karmaşıklığı önler. Ayrıca veri boyutunda azalmaya neden olduğu için depolama alanından da kazanç elde etmeye yarar.

2.2.1. Özellik seçimi adımları

Kullanışlı olabilecek özellikler seçilirken birkaç adım uygulanır. Öncelikle verinin ham halinden bir grup özellik ele alınarak değerlendirilir ve seçime uygun olup olmadıklarına karar verilir. Eğer bir özellik seçilmeye değer bulunursa sonuç kümesine dahil edilir ve algoritma tarafından belirlenmiş durdurma kriterine ulaşılana kadar bu işlem devam eder (Şekil 2.9).



Şekil 2.9. Özellik seçimi genel akış şeması (Yu, 2005)

2.2.1.1. Alt küme üretimi

Esas olarak alt küme oluşturma bir sezgisel arama sürecidir. Arama alanındaki her durum, değerlendirme için bir aday altküme belirler. Bu sürecin niteliği iki temel konu tarafından belirlenir. Birincisi, arama yönünü belirleyen arama başlangıç noktasına karar verilmelidir. Arama, boş bir kümeyle başlayabilir ve art arda özellikler ekleyebilir veya tam bir kümeyle başlayıp art arda özellikleri kaldırabilir veya her iki uçla başlayıp eşzamanlı olarak özellikleri ekleyip kaldırabilir. Arama, yerel çözüme takılmamak için rastgele seçilen bir alt kümeyle de başlayabilir.

İkinci olarak, bir arama stratejisine karar verilmelidir. N adet özellikli bir veri kümesi için 2^N kadar aday alt küme vardır. Bu sayı katlanarak arttıkça arama engellenebilir düzeye bile gelebilecektir. Bu nedenle, komple arama, ardışık arama ve rastgele arama gibi farklı yöntemler geliştirilmiştir.

- Komple Arama

Kullanılan değerlendirme kriterine göre en uygun sonucu bulmayı garanti eder. Tüm iyi alt kümeler kaçırılmadan kapsamlı bir arama tamamlandığında, tamlığın garanti edilmesi için bir aramanın ayrıntılı olması gerekmez. Optimum sonuç bulma şansını tehlikeye atmadan arama alanını azaltmak için farklı sezgisel fonksiyonlar kullanılabilir. Bunlara örnek olarak dal ve sınır ve ışın araması verilebilir. Böylelikle arama alanının büyüklüğüne rağmen, daha az sayıda alt grup değerlendirilir.

- Ardışık Arama

Ardışık arama işleminde veri setindeki tüm alt kümeler kullanılmamaktadır. Bu nedenle, en iyi alt kümeyi bulmayı garanti etmez. Optimum sonucu bulabilmek için ileri, geri ve çift yönlü yaklaşımlar kullanılmaktadır. Tüm bu yaklaşımlar, özellikleri birer birer ekler veya çıkarır.

- Rastgele Arama

Rastgele seçilen bir alt küme ile başlar ve iki farklı yolla ilerler. Bunlardan biri, klasik ardışık yaklaşımlara rasgelelik ekleyen sıralı bir araştırmayı izlemektir. Benzetilmiş tavlama bu yola örnek verilebilir. Diğeri, bir sonraki altkümeyi tamamen rastgele bir şekilde oluşturur. Las Vegas algoritması buna örnektir. Tüm bu yaklaşımlar için, rasgelelik kullanımı, arama alanındaki yerel çözümlerden kaçmaya yardımcı olur ve seçilen alt kümenin iyiliği, mevcut kaynaklara bağlıdır.

2.2.1.2. Alt küme değerlendirme

Yeni oluşturulan her bir alt kümenin bir değerlendirme kriteri ile değerlendirilmesi gerekir. Bir alt kümenin iyiliği her zaman belirli bir ölçüt tarafından belirlenir. Bir değerlendirme kriteri, sonunda seçilen özellik alt kümesinde uygulanacak olan madencilik algoritmalarına bağımlılıklarına bağlı olarak geniş bir şekilde iki gruba ayrılabilir. İki grup değerlendirme kriterini aşağıda tartışıyoruz.

Bağımsız Kriter: Tipik olarak, filtreleme modeli algoritmalarında bağımsız bir kriter kullanılır. Herhangi bir madencilik algoritmasına dahil olmadan eğitim verilerinin içsel özelliklerinden yararlanarak bir özellik veya özellik alt kümesinin iyiliğini değerlendirmeye çalışır. Uzaklık, bilgi, bağımlılık ve tutarlılık ölçütü sıklıkla kullanılan bağımsız kriter örnekleridir.

Bağımlı Kriter: Sarmal modellerde kullanılan bağımlı kriter, özellik seçiminde önceden belirlenmiş bir madencilik algoritması gerektirir ve hangi özelliklerin seçildiğini belirlemek için seçilen alt kümeye uygulanan bu algoritmasının performansını kullanır. Önceden

belirlenmiş madencilik algoritması daha uygun özellikler bulduğu için genellikle üstün performans verir, ancak aynı zamanda hesaplama açısından daha maliyetli olma eğilimindedir ve diğer madencilik algoritmaları için uygun olmayabilir. Örneğin, bir sınıflandırma probleminde, tahmini doğruluk birincil ölçüt olarak yaygın şekilde kullanılır. Özellik seçimi için bağımlı bir kriter olarak kullanılabilir. Daha sonra bu seçili özellikleri görünmeyen örneklerin sınıf etiketlerini tahminde kullanan sınıflayıcı tarafından özellikler seçildiğinden, doğruluk normalde yüksektir. Ancak her özellik alt kümesinin doğruluğunu tahmin etmek oldukça maliyetlidir.

2.2.1.3. Durdurma kriterleri

Durma kriteri, özellik seçimi işleminin ne zaman bitmesi gerektiğini belirler. Sık kullanılan bazı durma kriterleri şunlardır:

- Arama işleminin bitmiş olması,
- Minimum özellik sayısı veya maksimum iterasyon sayısı gibi belirlenmiş bazı sınırlara ulaşılmış olması,
- Herhangi bir özellik kümeye eklendiğinde veya çıkarıldığında daha iyi bir alt küme elde edilmemesi,
- Beklenen kriterleri sağlayan yeterince iyi bir alt kümeye ulaşılması.

2.2.1.4. Sonuç doğrulama

Sonuç doğrulama için en basit yol, veriler hakkında önceden bilinen bilgileri kullanarak sonucu doğrudan ölçmektir. Yapay verilerde olduğu gibi ilgili özellikleri önceden biliyorsak, bu bilinen özellikler kümesini seçilen özelliklerle karşılaştırabiliriz. Alakasız veya gereksiz nitelikte özellikler hakkındaki bilgi de yardımcı olabilir. Bu özelliklerin seçilmeleri beklenmez. Ancak gerçek dünyadaki uygulamalarda genellikle önceden böyle bir bilgiye sahip olunmamaktadır. Bu nedenle, madencilik performansındaki değişimi, özelliklerin değişimi ile izleyerek bazı dolaylı yöntemlere güvenmek zorundayız. Bu yöntemlerde, seçim başarımının belirlemesi için genellikle sınıflandırma hata oranı tercih edilmektedir.

2.2.2. Özellik seçimi yöntemleri

Özellik seçiminde birçok farklı yöntem kullanılmaktadır. Bu yöntemler, istatistiksel bilgiye dayalı filtreleme yöntemleri, özellik kümesinde arama yapan sarmal yöntemler ve bu iki yöntemin avantajlarını birleştiren hibrit yöntemler olarak kategorize edilebilir.

Özellik seçimi, filtreleme yöntemlerinde veri madenciliği algoritması çalışmadan önce yapılır. Bu algoritma sarmal yöntemlerde istenen özelliklerin seçimini yapmak için kullanılır. Hibrit yöntemlerde, madencilik ve seçim algoritmaları birlikte çalışır.

2.2.2.1. Filtreleme algoritması

Belirli bir veri seti için, algoritma önceden verilen boş, dolu veya rastgele seçilmiş bir alt kümeden aramaya başlar ve belirli bir arama stratejisi ile özellik alanı içinde arama yapar. Üretilen her bir alt küme, bağımsız bir ölçüt ile değerlendirilir ve öncekiyle karşılaştırılır. Daha iyi olduğu tespit edilirse, mevcut en iyi alt küme olarak kabul edilir. Arama önceden tanımlanmış bir durdurma kriterine ulaşana kadar tekrar eder. Algoritma, o andaki son en iyi alt kümeyi çıktı olarak verir. Arama stratejilerini veya değerlendirme ölçütlerini çeşitlendirerek farklı filtreleme algoritmaları tasarlanabilir.

Filtre modelleriyle özellik seçimi yapmak için, bir özelliğin sınıflandırma süreciyle olan ilişkisini ölçmek amacıyla birkaç farklı ölçüt kullanılır. Tipik olarak, bu ölçütler özellik değerlerinin, özniteliğin farklı aralıkları üzerindeki dengesizliğini hesaplar. Bazı ölçüt örnekleri aşağıdaki gibidir:

- Gini Index: $p_1 \ldots p_k$, ayrık özniteliğin belirli bir değerine karşılık gelen sınıfların kesri olsun. Daha sonra, ayrık özniteliğin bu değerinin gini endeksi şu şekilde hesaplanır:

$$G = 1 - \sum_{i=1}^{k} {p_i}^2 \tag{2.24}$$

G değeri 0 ile $1 - 1/k$ arasında değişmektedir. Küçük değerler sınıf dengesizliğinin göstergesidir. Bu, özellik değerinin sınıflandırma için daha ayırt edici olduğunu gösterir. Özelliğin genel gini endeksi, farklı özelliğin farklı değerlerine göre ağırlıklı ortalama alınarak

veya farklı farklı değerlerin herhangi birinin üzerinde maksimum gini endeksi kullanılarak ölçülebilir. Ağırlıklı ortalama daha yaygın olarak kullanılsa da farklı senaryolar için farklı stratejiler mevcuttur.

- Entropy: Ayrık özniteliğin belirli bir değerinin entropisi şu şekilde ölçülür:

$$E = -\sum_{i=1}^{k} p_i \log(p_i) \tag{2.25}$$

Gini-endeksi için olduğu gibi yukarıda da aynı gösterimler kullanılmıştır. Entropinin değeri 0 ile $\log(k)$ arasındadır, daha küçük değerler sınıf çarpıklığının göstergesidir.

- Fisher's Index: Bu endeks, sınıflar arası dağılımın sınıf içi dağılımına oranını ölçer. Fisher skoru aşağıdaki gibi hesaplanabilir:

$$F = \frac{\sum_{j=1}^{k} p_j(\mu_j - \mu)^2}{\sum_{j=1}^{k} p_j \sigma_j^2} \tag{2.26}$$

Eğer p_j, j sınıfına ait eğitim örneklerinin kesriyse, μ_j, j sınıfı için belirli bir özelliğin ortalamasıdır, μ, bu özellik için genel ortalamadır ve σ_j, bu özelliğin j sınıfı için standart sapmasıdır.

Filtre modeli herhangi bir madencilik algoritması içermeyen bağımsız değerlendirme kriterleri uyguladığından, madencilik algoritmasının herhangi bir dezavantajını bulundurmaz ve aynı zamanda hesaplama açısından verimlidir. Yaygın olarak kullanılan filtreleme yöntemlerinden bazıları; Fisher skor, Ki-Kare testi, Relief, bilgi kazancı ve kazanç oranıdır.

2.2.2.2. Sarmal algoritma

Genelleştirilmiş bir sarmal algoritma, alt küme değerlendirmesi için bağımsız bir ölçüt yerine önceden tanımlanmış bir madencilik algoritması kullanması dışında filtreleme algoritmasına çok benzer. Üretilen her alt küme için, özellik kümesine sahip veriye bir madencilik algoritmasını uygulayarak elde edilen sonuçların kalitesini değerlendirir. Bu nedenle, farklı madencilik algoritmaları farklı özellik seçim sonuçları üretecektir. Arama

stratejilerinin fonksiyon üretme ve madencilik algoritmaları aracılığıyla çeşitlendirilmesi farklı sarmal algoritmaların oluşmasına neden olabilir.

Sarmal yöntem, madencilik algoritmaları özellik alt kümelerinin seçimini kontrol etmek için kullanıldığından, önceden belirlenmiş madencilik algoritması için daha uygun özellik alt kümeleri bulunduğunda, üstün performans gösterme eğilimindedir. Fakat hesaplama maliyeti filtre modelinden fazladır. Ardışık ileri-geri yönde ve kayan seçim gibi çeşitleri mevcuttur.

2.2.2.3. Hibrit algoritma

Filtre ve sarmal modellerin avantajlarından yararlanmak ve durma kriterinin önceden belirlenme zorunluluğundan kaçınmak için, büyük veri setlerini ele almak üzere hibrit model önerilmiştir. Tipik bir hibrit algoritma, özellik alt kümelerini değerlendirmek için hem bağımsız bir ölçüt hem de madencilik algoritmasından yararlanır.

Belirli bir nicelik için en iyi alt kümelere karar verme amacıyla bağımsız ölçütleri ve farklı niceliklerdeki iyi alt kümeler arasından en iyi olanı seçmek için madencilik algoritmasını kullanır. Her turda, nitelikli olan en iyi alt küme için, kalan tüm özelliklerden bir özellik ekleyerek tüm olası daha nitelikli alt kümelerini araştırır. Yeni oluşturulan her bir alt küme bağımsız bir ölçütle değerlendirilir ve öncekilerle karşılaştırılır. Eğer daha iyiyse en iyi alt küme o olur. Her yinelemenin sonunda, bir madencilik algoritması en iyi seviyede uygulanır ve elde edilen sonucun kalitesi, önceki en iyi seviyedeki alt kümeyle karşılaştırılır. Eğer kalite daha iyiyse, algoritma bir sonraki seviyede en iyi alt kümeyi bulmaya devam eder. Aksi takdirde, durur ve geçerli en iyi alt kümeyi son en iyi alt küme olarak çıkarır. Bir madencilik algoritmasından elde edilen sonuçların kalitesi, hibrit modelde doğal bir durma kriteri sağlar.

2.3. Sınıflandırma Algoritmaları

Veri sınıflandırma problemi, çok çeşitli veri madenciliği alanlarında sayısız uygulamaya sahiptir. Bunun nedeni, problemin bir dizi özellik değişkeni ve ilgilenilen bir hedef değişken arasındaki ilişkiyi öğrenmeye çalışmasıdır. Pratikte birçok sorun, özellik ve hedef değişkenler arasındaki ilişki olarak ifade edilebildiğinden, bu model için geniş bir uygulanabilirlik sağlar. Sınıflandırma kavramı, basitçe bir veri seti üzerinde tanımlı olan çeşitli sınıflar arasında veriyi

dağıtmaktır. Sınıflandırma algoritmaları, verilen eğitim kümesinden bu dağılım şeklini öğrenirler ve daha sonra sınıfının belirli olmadığı test verileri geldiğinde doğru şekilde sınıflandırmaya çalışırlar.

Sınıflandırma algoritmaları tipik olarak iki aşamadan oluşur. Eğitim örneklerinden bir modelin oluşturulduğu eğitim aşaması ve etiketlenmemiş bir test örneğine etiket atamak için kullanılan test aşamasıdır. Veri kümesi üzerinde verilen bu sınıfları belirten değerlere etiket ismi verilir ve gerek eğitim gerekse test sırasında verinin sınıfının belirlenmesi için kullanılırlar. Bazı durumlarda, eğitim aşaması tamamen ihmal edilir ve sınıflandırma, eğitim örneklerinin test örnekleri ile ilişkisinden doğrudan gerçekleştirilir. En yakın komşu sınıflandırıcıları gibi örneğe dayalı yöntemler böyle bir senaryoya örnektir. Bu gibi durumlarda bile, test aşamasında verimi sağlamak için bir ön işleme aşaması gerçekleştirilebilir.

Bir sınıflandırma algoritmasının çıktısı, iki yoldan biriyle sunulabilir. Birinde test örneği için direkt bir etiket bulunur. Diğerinde ise, her sınıf etiketi ve test örneği birleşimi için sayısal bir puan döndürülür. Sayısal puan, bir test örneği için en yüksek puana sahip olan sınıf seçilerek ayrı bir etikete dönüştürülebilir. Bu puanlamanın avantajı, farklı test örneklerinin belirli bir önem sınıfına ait olma eğilimini karşılaştırmayı ve gerektiğinde bunları sıralamayı mümkün hale getirmesidir.

Bilginin sınıflandırılması, karar verme işleminin önemli bir bileşenidir. Bu işlemlerin çoğu, sınıflandırma probleminin örnekleridir veya tahmin, teşhis ve örüntü tanıma gibi uygulamalar bir sınıflandırma problemine kolayca dönüştürülebilirler. Sınıflandırma işlemi, internetin ortaya çıkması ile daha da büyük önem kazanmıştır. Bir iletişim ve işlem kanalı olarak internet, işbirlikçi filtreleme ve öneri sistemleri gibi birçok yeni özellik sağlayan teknolojiyi uygulamak için ortam sağlar. Tavsiye sistemleri, tüketicilere çevrimiçi olarak mevcut ürünler ve bilgiler hakkında önerilerde bulunarak yardımcı olmayı amaçlamaktadır. Bu sistemlerin genel amacı, mevcut bilgileri bazı kriterlere göre sınıflandırmak ve kullanıcılara hangi önerilerde bulunulacağına karar vermektir. Hastalık teşhisi, müşteri kitlesi tespiti, belge kategorizasyonu, sosyal ağ ve multimedya verilerinin analizi gibi birçok alanda sınıflandırma uygulamaları yapılmaktadır.

Lineer sınıflandırıcılar, en yakın komşuluk, destek vektör makineleri, karar ağaçları ve yapay sinir ağları makine öğrenmesinde kullanılan bazı sınıflandırma algoritmalarıdır.

2.3.1. Naive Bayes Sınıflandırıcısı

Naive Bayes sınıflandırıcısı, tahminler arasında bağımsızlık varsayımıyla İngiliz matematikçi Thomas Bayes'in adını verdiği Bayes Teoremine dayanan bir sınıflandırma tekniğidir. Basit bir ifadeyle, bir sınıftaki belirli bir özelliğin varlığının diğer herhangi bir özelliğin varlığı ile ilgisiz olduğunu varsayar. Bu özellikler birbirine veya diğer özelliklerin varlığına bağlı olsa bile, bu özelliklerin tümü bağımsız olarak olasılıklara katkıda bulunur. Naive Bayes modelinin oluşturulması kolaydır ve özellikle çok büyük veri kümeleri için kullanışlıdır. Sadeliği ile birlikte, Naive Bayes'in oldukça karmaşık sınıflandırma yöntemlerinden bile daha iyi performans gösterdiği bilinmektedir.

Bayes teoremi, koşullu olasılık hesaplamaları için bir yol sağlar. Naive Bayes sınıflandırıcısı, bir prediktörün değerinin verilen bir sınıf üzerindeki etkisinin, diğer prediktörlerin değerlerinden bağımsız olduğunu varsayar. Bu varsayıma sınıf şartlı bağımsızlık denir.

$$P(c|x) = \frac{P(C|X)P(c)}{P(x)} \tag{2.27}$$

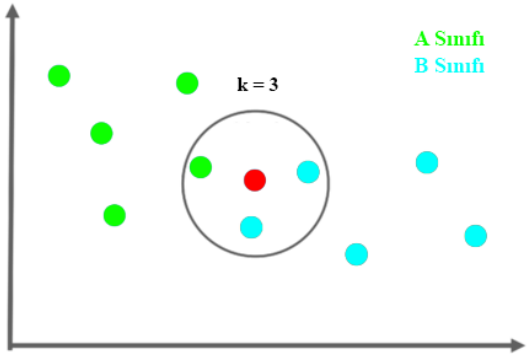
$$P(c|x) = P(x_1|c) \times P(x_1|c) \times \dots \times P(x_n|c) \times P(c) \tag{2.28}$$

- $P(c|x)$ Bir sınıfın verilen prediktör için sonsal olasılık dağılımı
- $P(c)$ Parametre veya parametre vektörü için önsel olasılık dağılımı
- $P(x|c)$ Verilen sınıfın olabilirlik fonksiyonu
- $P(x)$ Prediktörün önsel olasılığı

2.3.2. K-En Yakın Komşuluk

En yakın komşuluk (KNN) algoritmaları, 1967 yılında T. M. Cover ve P. E. Hart tarafından önerilmiş bir sınıflandırma algoritmasıdır. Birçok etiketli noktayı alır ve diğer noktaları nasıl etiketleyeceğini öğrenmek için bunları kullanır. Yeni bir noktayı etiketlemek için, o yeni noktaya en yakın olan etiketli k adet noktaya bakar ve bu komşuların etiketleri kullanır. Bu nedenle komşularının en fazla olan etiketi, yeni noktanın etiketidir.

KNN basit ve gürültülü eğitim verilerine karşı güçlü olması sebebiyle en yaygın kullanılan makine öğrenme algoritmalarından biridir. Fakat uzaklık hesabı yaparken bütün durumları sakladığından, büyük veriler için kullanıldığında fazla miktarda bellek alanına ihtiyaç duymaktadır.



Şekil 2.10. En yakın nokta komşuluğuna göre sınıf tespiti

Komşuluk durumu (Şekil 2.10) belirlenirken bir noktanın diğer noktalar ile olan uzaklığına bakılır. Uzaklık hesapları için genelde 3 farklı uzaklık fonksiyonu kullanılmaktadır:

- Euclidean Uzaklığı

$$d(x,y) = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{2.29}$$

- Manhattan Uzaklığı

$$d(x,y) = \sum_{i=1}^{k}|x_i - y_i| \tag{2.30}$$

- Minkowski Uzaklığı

$$d(x,y) = \left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{\frac{1}{q}} \tag{2.31}$$

En yakın komşuluk algoritması çalışırken ilk olarak verilen bir noktaya en yakın komşuların sayısı olan k parametresi belirlenir. Bu sayıya göre komşular bulunur ve sınıflandırma yapılır. Veriye eklenecek olan yeni değerin uzaklık fonksiyonları yardımıyla diğer verilere uzaklığı hesaplanır. İlgili uzaklıklardan en yakın komşular ele alınır. Özniteliklerine bakarak yeni veri komşularının sınıfına atanır ve etiketlenmiş olur.

2.3.3. Karar Ağaçları

Karar ağacı, bir ağaç yapısı şeklinde sınıflandırma veya regresyon modelleri oluşturur. Bir veri kümesini daha küçük alt gruplara ayırırken, aynı zamanda ilişkili bir karar ağacı adım adım geliştirilir. Sonuç, karar düğümleri ve yaprak düğümleri olan bir ağaçtır. Bir karar düğümünün iki veya daha fazla dalı vardır ve bir yaprak düğümü bir sınıflandırma veya kararı temsil eder. Ağaçtaki en üst karar düğümü, kök düğümü adı verilen en iyi tahmine karşılık gelir. Karar ağaçları hem kategorik hem de sayısal verileri ele alabilir.

- **Rastgele Orman:** Rastgele ormanlar veya rassal karar ormanları, sınıflandırma, regresyon ve diğer görevler için, eğitim zamanında çok sayıda karar ağacı oluşturarak ve sınıfların modu veya tek tek ağaçların ortalama tahmini olan sınıfı çıkaran işleyen bir topluluk öğrenme yöntemidir.

- **Hızlandırılmış Ağaçlar:** Hem sınıflandırma hem de ilkelleme problemleri için kullanılabilen bir algoritmadır.

- **Döndürme Ağacı:** Rastgele ağaca benzer şekilde birden fazla ağaç kullanılmaktadır. Fakat her ağaç, önce farklı bileşen analizi (PCA) kullanılarak eğitilmektedir. Bu eğitim için veri kümesinin rastgele seçilmiş bir alt kümesi kullanılmaktadır.

- Ayrıca ID3 ve C.45 gibi algoritmalar da karar ağacı öğrenmesinde kullanılmaktadır.

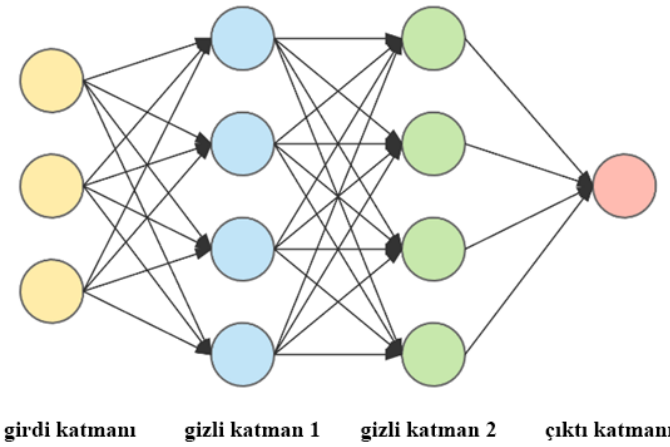
2.3.4. Destek Vektör Makineleri

Sınıflandırma için bir düzlemde bulunan etiketli gruplar arasında bir sınır çizilerek bu grupları ayırmak mümkündür. Düzlemleri ayıracak bu karar sınırının çizileceği yer ise grup üyelerine en uzak olan yer olmalıdır. Destek Vektör Makineleri (DVM) bu sınırları belirler. Bu yöntem 1995 yılında Vladir Vapnik, Berhard Boser ve Isabelle Guyon tarafından geliştirilmiştir. Günümüzde DVM yüz tanıma sistemlerinden metin kategorizasyonuna kadar birçok sınıflandırma probleminde kullanılmaktadır.

2.3.5. Yapay Sinir Ağları

Yapay sinir ağları, canlılarda bulunan sinir sistemi hücrelerinin çalışma biçiminden esinlenerek geliştirilmiştir. Amacı canlı beyninin öğrenme yeteneğini bilgisayarlara kazandırabilmektir. İlk olarak 1943 yılında bir nörobiyolog olan W.S. McCulloch ve matematikçi W.A. Pitts tarafından incelenmiştir. Sonrasında birçok bilim insanının katkılarıyla yapay sinir ağları üzerine çalışmalar yapılarak geliştirilmiştir.

Bir sinir ağı, bir girdi vektörünü bir çıktıya dönüştüren, katmanlar halinde düzenlenmiş birimlerden (nöronlar) oluşur. Her birim bir girdi alır, ona genellikle doğrusal olmayan bir fonksiyon uygular ve ardından çıktıyı bir sonraki katmana iletir. Genellikle ağlar ileriye doğru beslenecek şekilde tanımlanır. Bir birim çıktısını bir sonraki katmandaki tüm birimlere besler, ancak önceki katmana geri bildirim iletmez. Ağırlıklar bir birimden diğerine geçen sinyallere uygulanır ve bunlar yapay sinir ağını eldeki belirli soruna uyarlamak için eğitim aşamasında ayarlanan ağırlıklardır.



Şekil 2.11. Yapay sinir ağları genel yapısı

Yapay sinir ağlarının günümüzde en yaygın olarak kullanılan modeli çok katmanlı algılayıcı ağlarıdır. Çok katmanlı yapay sinirleri temelde 3 kısımdan (Şekil 2.11) oluşmaktadır. Girdi katmanında bilgi işleme gerçekleşmez. Bilgiler alınıp gizli katmanlara iletilir. Girdi katmanındaki her bir eleman, gizli katmanındaki işlem elemanlarının tümüne bağlıdır. Bu kısımda girdi katmanından gelen bilgiler işlenir. Bir adet gizli katman ile birçok problemi çözmek mümkündür. Fakat birden fazla gizli katmanda kullanılabilmektedir. Problemin türüne

göre gizli katmaların sayısı değişmektedir. Çıktı katmanı ise gizli katmandan gelen bilgileri işleyerek dışarı iletir.

$$f(x) = b + \sum_{i=1}^{n}(x_i\, w_i) \tag{2.32}$$

Burada:

b = eğilim, x = nöron girdisi, w = ağırlıklar, n = gelen katman girdi sayısı, i = 0'dan n'e sayaç

Yapay sinir ağlarında girdilerin aldığı değerler bağlantıların ağırlıkları ile çarpılır ve sonuçlar birleştirilerek ağın net girdisi bulunur. Net girdiler aktivasyon fonksiyonuna sokulduktan sonra ağın net çıktısı elde edilmiş olur.

2.3.6. Sınıflandırmada kullanılan metrikler

Hedef verisi önceden belli olan veri setlerinden elde edilen modellerin performansını değerlendirmek üzere en sık kullanılan yöntem karışıklık matrisidir. Modelin, pozitif ve negatif örnekleri içerisinde barındıran test veri setini ne ölçüde sınıflandırdığını gösteren matrise karışıklık matrisi (confusion matrix) denir (Çizelge 2.3).

Çizelge 2.3. Karışıklık matrisi

		Tahmin Sınıfı	
		Pozitif	Negatif
Gerçek Sınıf	Pozitif	TP	FN
	Negatif	FP	TN

- TP (Doğru Pozitif): Test verisindeki değer ile modelin tahmin ettiği sınıfın aynı olduğu durumda kullanılır. Doğru sınıflandırma yapılmıştır.
- FN (Yanlış Negatif): Test verisindeki değer ile modelin ürettiği sınıf farklı bulunmuş pozitif ilken negatif sınıflandırılmıştır. Sınıflandırma hatalıdır.
- FP (Yanlış Pozitif): Gerçek değer negatif iken hatalı şekilde pozitif sınıflandırılmıştır.
- TN (Doğru Negatif): Değer negatif iken doğru biçimde negatif sınıflandırılmıştır.

Doğruluk oranı: Sınıflandırma başarımının ölçümünde en yaygın kullanılan yöntem doğruluk oranıdır (accuracy rate). Doğru sınıflandırılmış örneklerin toplam örnek sayısına oranlanması ile oluşur.

$$A = \frac{(TP+TN)}{(TP+FP+FN+TN)} \tag{2.33}$$

Hata oranı: Yanlış sınıflandırılmış örnek sayısının toplam sayıya bölümüdür. Doğruluk oranının 1'e tamlayanı $(1 - A)$ olarak da ifade edilebilir.

Kesinlik: Kesinlik (precision), sınıfı pozitif olarak tahmin edilen TP sayısının, sınıfı 1 olarak tahmin edilmiş örnek sayısına oranıdır.

$$P = \frac{TP}{(TP+FP)} \tag{2.34}$$

Duyarlılık: Tüm pozitif sınıflardan, ne kadar doğru tahmin edildiğini belirten metrik, duyarlılık (recall) olarak tanımlanmalıdır.

$$R = \frac{TP}{(TP+FN)} \tag{2.35}$$

F-Ölçütü: Duyarlılık ve kesinlik ölçütlerinin anlamlı sonuç üretmeye yeterli olmadığı durumlarda, bu iki ölçütü birlikte değerlendirmek gerekmektedir. Bu nedenle f-ölçütü (f-measure) tanımlanmıştır. Bu ölçüt, kesinlik ve duyarlılığın harmonik ortalamasıdır.

$$F = \frac{2RP}{(R+P)} \tag{2.36}$$

Kappa istatistiği: Temelde, Kappa istatistiği, makine öğrenmesi sınıflandırıcısı tarafından sınıflandırılan örneklerin, zemin doğruluğu olarak etiketlenen verileri ne kadar yakından eşleştirdiğini ve beklenen doğruluk ile ölçülen rastgele bir sınıflandırıcının doğruluğunu kontrol eden bir ölçüdür. P_o kabul edilen oran, P_c kabul edilmesi beklenen oran olmak üzere, Kappa değeri şu şekilde hesaplanır:

3. MATERYAL VE YÖNTEM

3.1. Müzik Veri Seti

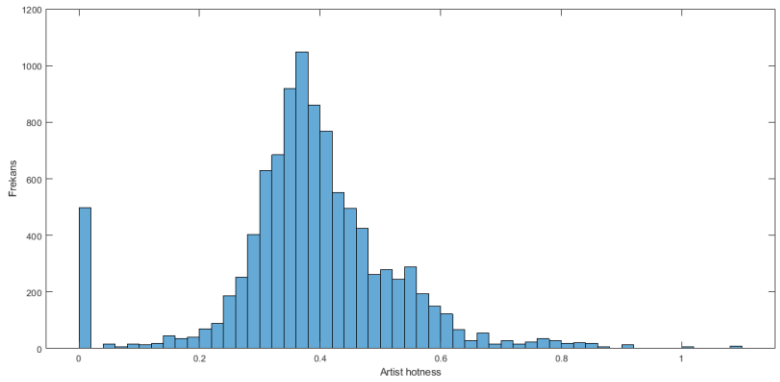
Bu araştırmada kullanılan açık kütüphane yaklaşık bir milyon adet çağdaş ve popüler şarkı hakkında veri elde etmek amacıyla Echo Nest şirketi tarafından akıllı makine dinlemeye yönelik bir laboratuvar olan LabROSA'nın katkılarıyla Million Song Dataset adı altında oluşturulmuştur. Bu proje ayrıca müzik bilgi edinme alanında daha fazla araştırma yapılmasını teşvik etmek için geniş bir veri seti sağlamak amacıyla Amerika Ulusal Bilim Vakfı (NSF) tarafından finanse edilmiştir. Veriler, sanatçının adı, albüm ve yayınlanan yıl gibi şarkılar hakkında standart bilgiler içerir. Bunlara ek olarak şarkının uzunluğu, kaç tane müzikal bar sürdüğü ve solma süresi gibi daha gelişmiş bilgileri de bulundurur.

Million Song Dataset analiz edilerek veriler sınıflandırılacaktır. Orijinal veri seti 280GB boyutunda ve bir milyon parçadan oluşmaktadır. Bu çalışmada 10.000 şarkıdan oluşan bir subset kullanılacaktır. Böylece veri büyüklüğünden kaynaklanan sorunlardan kaçınılmıştır. Boyutu küçültülmüş veri seti kümesi sanatçı adı, başlık, süre ve tempo gibi alanlar içeren 22 öznitelikten oluşmaktadır.

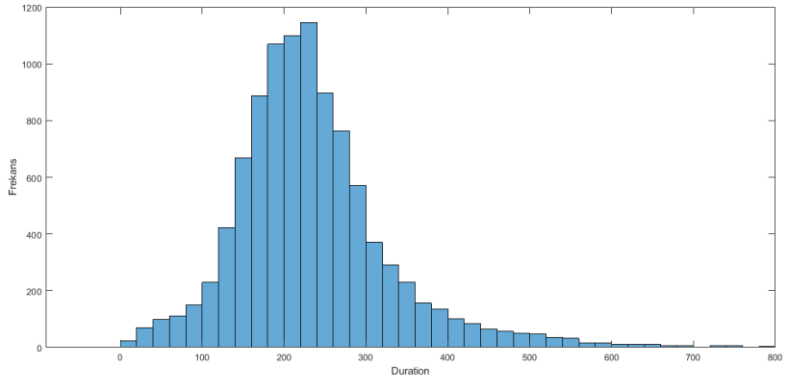
Orijinal veri setinde, bir şarkı popülarite özelliği bulunmaktadır. Fakat bu özellikte veri girdi sayısının neredeyse yarısı olan yaklaşık 4500 şarkının değerleri girilmemiştir. Bu nedenle popülerliği belirlemek için BillBoard Top 100 listesi kullanılmıştır. Bir şarkı en az bir kez BillBoard Top 100'e çıkarsa, hit parça olarak tanımlanır. Veri setinde bulunan 10.000 şarkıdan 1192 parça, popüler şarkı olarak sınıflandırılmıştır.

3.1.1. Veri görselleştirme

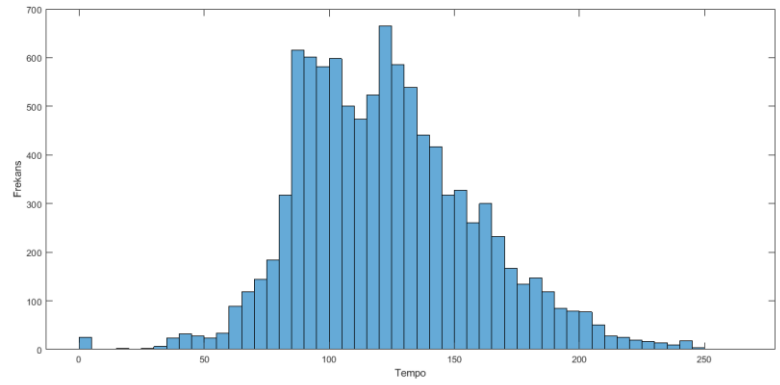
Veri setindeki farklı alanların dağılımı hakkında bir fikir edinmek için bazı grafikler çizdirilmiştir. Şekil 3.1'den görülebileceği gibi, tempo, sanatçı popülariteleri, şarkıların süresi gibi temel alanlar destek vektör makineleri, kNN gibi mesafe ölçüm tabanlı modeller için esas olan Gauss dağılımına uymaktadır. Parçaların çıktığı yıllara ait verilerin bu dağılıma uymadığı gözlenmiştir.



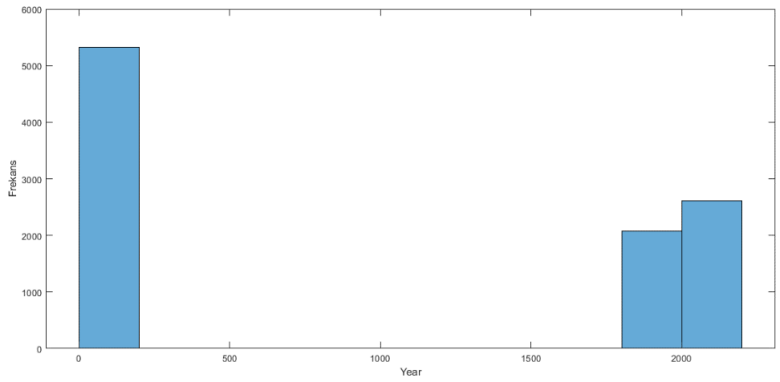
Şekil 3.1. a) Sanatçı popülaritesine ait histogram grafiği



Şekil 3.1. b) Parça süresine ait histogram grafiği

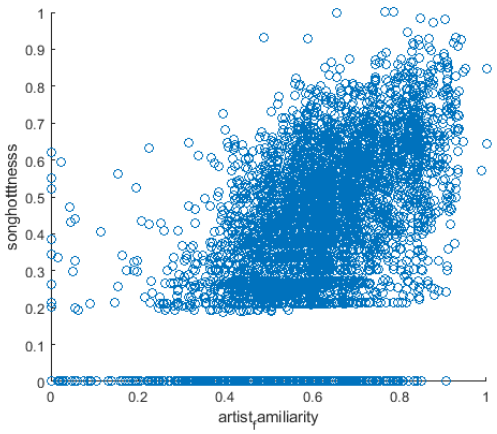


Şekil 3.1. c) Şarkı temposuna ait histogram grafiği

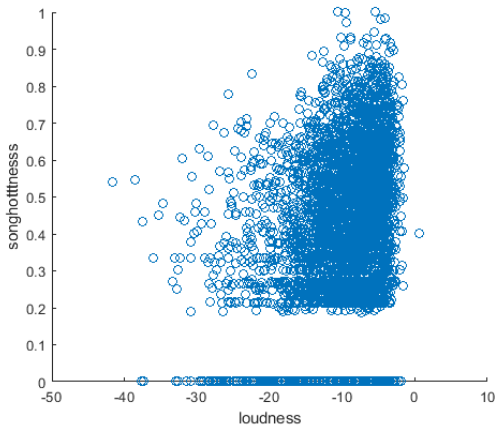


Şekil 3.1. d) Yıl verisine ait histogram grafiği

Veri kümesindeki belirli alanlar ile parça popülaritesi arasındaki korelasyonu incelemek için dağılım grafikleri çizdirilmiştir.



Şekil 3.2. Sanatçı benzerliği-popülarite dağılım grafiği



Şekil 3.3. Ses yüksekliği-popülarite dağılım grafiği

Şekil 3.2 ve Şekil 3.3'ten de anlaşılacağı gibi, sanatçı benzerlikleri ve şarkı ses yükseklikleri, şarkının popülerliği ile ilişkilidir. Sanatçının benzerliği, beklendiği gibi pozitif korelasyon göstermektedir. Ancak şaşırtıcı bir şekilde, şarkı gürültüsü, popülerlik ile negatif korelasyon göstermektedir. Daha popüler şarkıların daha yüksek sesli olması beklenmiştir ancak genel olarak şarkıların ortalama ses yüksekliği biraz daha yüksek olduğu için bunun tam tersi görünmektedir. Şekil 3.3'de ses yüksekliği x ekseni üzerinde, sıcaklığı ise y üzerinde çizilmiştir. Daha popüler şarkıların daha sessiz olmasının sebebi, veride genel olarak şarkıların popülarite ortalamasını düşüren aşırı yüksek sesli şarkıların olması gibi görünmektedir.

3.1.2. Veri ön işleme

Müzik veri seti üzerinde veri ön işleme yapılarak daha verimli sonuçlar alınması amaçlanmıştır. Veride bulunan öznitelik kümesinden şarkı adı, sanatçı lokasyonu gibi metin içerikli veriler çıkarılarak, algoritmalar ile hesaplama yapabilmeye müsait olan nümerik veriler (Çizelge 3.1) ayrıştırılmıştır.

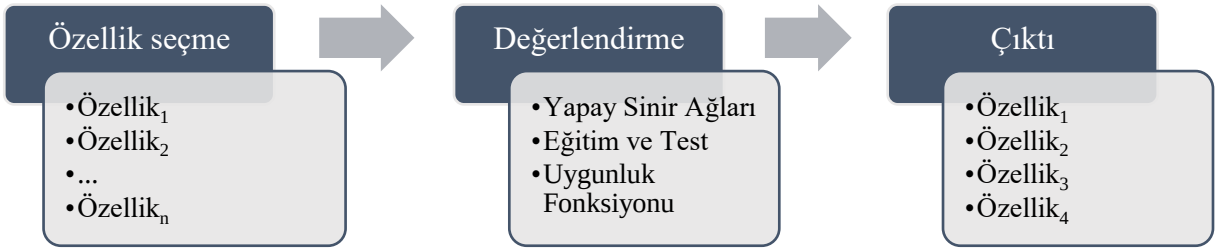
Çizelge 3.1. Nümerik veriler hakkında temel istatistikler

Özellik	Adet	Ortalama	Std. sapma	Min.	Maks.
artist_familiarity	9997	0.565	0.16	0	1
artist_hotttnesss	10001	0.386	0.144	0	1.083
artist_latitude	3742	37.157	15.599	-41.281	69.651
artist_longitude	3742	-63.934	50.508	-162.44	174.77
duration	10001	238.512	114.133	1.044	1819.8
end_of_fade_in	10001	0.759	1.868	0	43.119
key	10001	5.276	3.554	0	11
key_confidence	10001	0.45	0.275	0	1
loudness	10001	-10.485	5.4	-51.643	0.566
mode	10001	0.691	0.462	0	1
mode_confidence	10001	0.478	0.191	0	1
song_hotttnesss	5649	0.343	0.247	0	1
start_of_fade_out	10001	229.98	112.191	1.044	1813.4
tempo	10001	122.921	35.186	0	262.83
time_signature	10001	3.565	1.266	0	7
time_signature_confidence	10001	0.51	0.373	0	1
year	10001	935	996.651	0	2010

Parçaların yayınlandığı yıl verisini tutan “year” etiketine ait verilerde yılı bilinmeyen parçaların fazlalığı ve bu verilere 0 değeri girilmiş olması dengesiz bir dağılım oluşturmaktadır. Bu dağılımın hesaplamalarda sapmalara sebep olmaması istendiğinden yıl özellik sütunu da işlenen veriden çıkartılmıştır. Veri temizleme yapıldıktan sonra 16 adet nümerik özellik ve hedef sütunun bulunduğu bir veri seti elde edilmiştir. Son durumda data 16x10001 giriş ve 1x100001 hedeften oluşmaktadır. Bu bilgilere dayanarak üstünde özellik seçimi yapılacak virgülle ayırılmış değerler (.csv) dosyası kullanılacak yazılımlara uygun formatlara (.mat, .arff) dönüştürülmüştür.

3.2. Yöntem

Özellik seçme işlemleri istatistik, optimizasyon, nümerik analiz gibi pek çok matematiksel hesaplamanın etkili ve hızlı şekilde yapılmasına olanak sağlayan Matlab yazılımı kullanılarak tasarlanan algoritmalar ile yapılmıştır. Sınıflandırma performansını ölçmek amacıyla yapay sinir ağları kullanan meta-sezgisel yöntemler (KKA, PSO, BT ve GA) ile istenen özelliklerin çıkarımı yapılmıştır. Önerilen yöntemde, Şekil 3.4’de belirtilen akış şemasına uygun şekilde, öncelikle bir grup özellik seçilir. Doğa esinli algoritmalarda bu özellikler seyahat edilecek konumlar olarak modellenebilmektedir. Sonrasında seçilen veriler değerlendirilerek optimal sonuçlar döndürülür. Böylelikle seçilmesi istenen özellik kümesi elde edilmiş olur.



Şekil 3.4. Özellik seçimi adımları

Elde edilen özelliklerin sınıflandırma başarımını test etmek için farklı sınıflandırıcılar kullanılmak istenmiştir. İçerisinde birçok farklı kümeleme ve sınıflandırma algoritması barındıran, veri madenciliği uygulamalarına imkân veren, açık kaynak kodlu Weka 3 makine öğrenme yazılımı kullanılmıştır. Karar ağaçları, Naive Bayes ve kNN gibi sınıflandırıcılar kullanılarak seçilen özelliklerin başarımları ölçülmüştür.

3.2.1. Karınca Koloni Algoritması ile özellik seçimi

Girdi ve hedefleri içeren veri üzerinde özellik çıkarımı yaparak boyut azaltmak için karınca koloni algoritması kullanılmıştır. Uygulama aşamasında istenen özellik sayısı 4 olarak belirtilmiştir. Problem tanımlanmış ve uygunluk fonksiyonu oluşturulmuştur. Karınca koloni optimizasyonunda kullanılacak parametre değerleri (Çizelge 3.2) girilmiştir.

Çizelge 3.2. KKA parametre değerleri

Parametre	Değer
Karınca sayısı (populasyon)	50
İlk feromon değeri	1
Feromon iz bilgisi (alpha)	1
Sezgisel parametre (beta)	1
Buharlaşma oranı	0.05

Başlangıç aşamasında sezgisel bilgi ve feromon matrisleri ve en iyi uygunluk değerlerini tutan dizi oluşturulmuştur. İlk olarak rastgele başlangıç pozisyonları belirlenmiştir. Ana döngü yapay karıncaların istenen iterasyon sayısı kadar turlarını tamamlayacağı şekilde tasarlanmıştır. Bu modelde, özellikler karıncaların gezeceği lokasyonlardır. Her bir karıncanın izlediği yol verisi tutulmaktadır. Karıncalar turlarını tamamladıktan sonra elde edilen değerler yapay sinir ağlarına gönderilerek değerlendirilmektedir. Uygunluk fonksiyonu aracılığıyla her turun minimum hata değeri hesaplanarak kaydedilmektedir. Sonrasında buharlaşma oranı parametresine göre feromon güncellemesi yapılarak bir sonraki tekrara geçilmektedir. Eğer yeni turda daha iyi bir değer elde edilmişse, bu anlık çözüm olarak tanımlanmaktadır. Döngü istenen sayıda tekrarlandıktan sonra algoritma tamamlamakta ve bulunan en iyi çözüm gösterilmektedir.

Özellik seçimi işleminde öncelikle 50 adet karıncanın tur, maliyet ve çıktı değerlerini tutan boş bir matris oluşturulmaktadır. Başlangıçta belirlenen iterasyon sayısı kadar çalışacak olan döngüde ise ilk karıncadan başlamak üzere turun başlayacağı özellik rastgele biçimde seçilmektedir. Sonrasında bu karıncanın diğer özelliklere geçme olasılığı hesaplanmaktadır. Pozisyonlar feromon değerlerine göre rulet çarkı seçimine sokularak karıncanın gideceği bir sonraki özellik belirlenmektedir. Karınca turunu tamamladığında elde edilen değerler uygunluk fonksiyonuna gönderilerek maliyet değeri hesaplanmaktadır. Bir karıncanın yaptığı turdaki özelliklerin sırası, turun maliyet değeri ve çıktı olarak istenen sayıda özelliğin bulunduğu yapı

kaydedilmektedir. Bunun ardından karıncanın bıraktığı feromon değerleri güncellenmektedir. Döngü bir sonraki karıncaya geçmekte ve aynı işlemler tekrarlanmaktadır. Her iterasyon için 0,05 oranında feromon buharlaştırılmakta ve bulunan en iyi maliyet değeri kaydedilmektedir.

3.2.2. Parçacık Sürü Optimizasyonu ile özellik seçimi

Parçacık sürü optimizasyonu kullanarak özellik seçimi yapılırken öncelikle karar değişkenlerinin sayısı, alt ve üst sınırları ve bunları içeren matrisin boyutu belirlenmiştir. Algoritmada kullanılmak üzere maksimum iterasyon sayısı, popülasyon boyutu, atalet ağırlığı ve öğrenme katsayıları parametre olarak tanımlanmıştır.

Popülasyon boyutu toplamda 50 parçacıktan oluşmaktadır. Fi (phi) sabitlerinin değeri 2,05 alınmış ve bunların toplamı ki-kare yönteminden geçirilerek atalet ağırlığına eşitlenmiştir. Bu ağırlığın sönümleme oranı 0,99'dur. Bireysel ve sosyal öğrenme katsayıları (c1, c2), fi sabitleri ile ki-kare formülünden gelen değerin çarpılmasıyla bulunmuştur. Hız limitleri belirlenmiş ve minimum limit maksimum limitin negatifi olacak şekilde ayarlanmıştır.

$$chi = 2/(phi - 2 + sqrt(phi^2 - 4 \times phi)) \tag{3.1}$$

Başlangıç pozisyonu ve parçacık hızı tanımlanmıştır. Ana döngüde parçacıkların hız güncellemesi, hız limitinin uygulanması, yer değiştirme için pozisyon güncellemesi, belirlenen sınırların dışına çıkmamak için pozisyon limitlerinin belirlenmesi ve uygunluk fonksiyonu ile değerlendirme yapıldıktan sonra kişisel ve global en iyi değerlerin güncellemesi yapılmıştır. Sonuç olarak da en iyi maliyet değerleri döndürülmüştür.

İstenen özelliklerin seçilme işleminde, parçacık sürü optimizasyonu kullanılarak 4 adet özelliğin bulunması gerekmektedir. Değişken sayısı verideki sütun sayısından çekilmiştir. İterasyon adedi belirlendikten sonra parçacıkların pozisyon, maliyet, çıktı ve hızlarının tutulduğu tablo oluşturulmuştur. Değişkenlerin üst sınırı 1, alt sınırı 0 olarak belirlenmiştir. Başlangıçta, parçacık pozisyonunun belirlenmesi için değişken sayısı kadar değer üreten, alt ve üst sınırları parametre olarak alan sürekli tekdüze dağılım fonksiyonu kullanılmıştır. Parçacıkların başlangıç hızı sıfırdır. Üretilen uygunluk fonksiyonuna gönderilerek kişisel en iyi

pozisyon ve maliyet değerleri hesaplanmıştır. Eğer bulunan değer çözümün en küçük maliyetinden düşük ise global en iyi olarak tutulmaktadır.

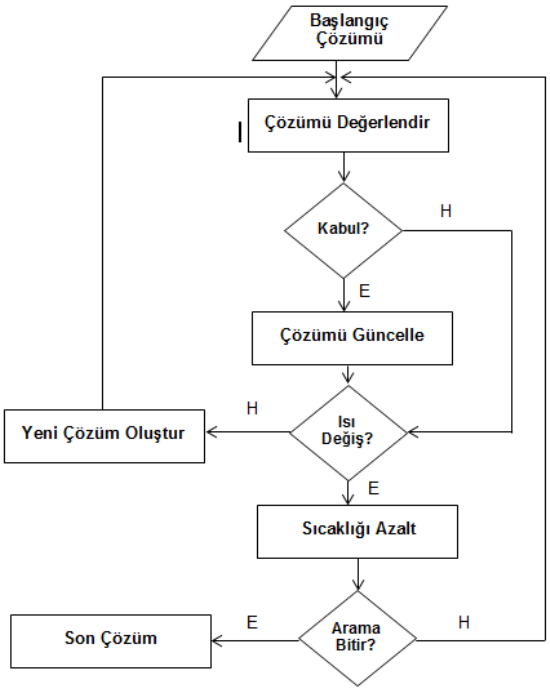
$$particle(i).Position = unifrnd(VarMin, VarMax, VarSize) \quad (3.2)$$

Toplam iterasyon sayısı kadar çalışması planlanan döngüde ise her parçacık için atalet ağırlığı (w) ve öğrenme sabitleri (c1, c2) kullanılarak hız güncellemesi yapılmıştır. Ardından maksimum ve minimum hız limitleri (V_{min}, V_{max}) uygulanmıştır. Parçacığın hızına ve önceki konumuna göre gideceği yeni pozisyon bulunmuştur. İstenen aralığın dışına çıkılmaması için pozisyon limitleri de uygulandıktan sonra parçacıklar değerlendirilip en iyi maliyet güncellenmiştir. Yeni atalet ağırlığı sönümleme katsayısı ile çarpılarak bulunmuş ve döngü sonlandırılmıştır. Bulunan iyi çözüm kaydedilmiş, iterasyon-maliyet grafiği çizdirilmiştir.

3.2.3. Benzetilmiş Tavlama ile özellik seçimi

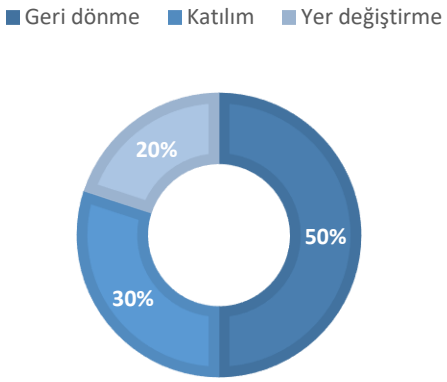
Problemin tanımı yapıldıktan sonra benzetilmiş tavlama algoritmasının parametreleri belirlenmiştir. Maksimum iterasyon ve alt iterasyon sayısı, başlangıç sıcaklığı ve sıcaklığın azalma oranı girilmiştir. Başlatma aşamasında başlangıç çözümü oluşturma ve değerlendirmesi yapılmış ve bulunan en iyi çözüm tanımlanmıştır. En iyi maliyet değerlerini tutan dizi oluşturulmuş ve başlangıç sıcaklığı belirlenmiştir.

Belirlenen iterasyon sayısı kadar çalışacak döngü içerisinde yeni çözüm, komşu üretme fonksiyonu kullanılarak oluşturulmuştur. Bu fonksiyonda yer değiştirme, geri dönme ve katılma işlemleri uygulanmaktadır. Üretilen yeni çözümler uygunluk fonksiyonu kullanılarak değerlendirilmektedir. Algoritmanın sonraki aşamasında, eğer yeni çözümün maliyet değeri genel çözüm maliyetinden düşük ise bu çözüm, global çözüm olarak atanmaktadır. Eğer değil ise algoritma daha iyi bir çözüm bulana kadar çalışmaya devam etmektedir. Döngü tamamlanmadan önce, bir iterasyonda bulunan en iyi çözüm maliyeti kaydedilmektedir. İterasyon bilgisi yazdırıldıktan sonra daha önce belirtilen soğuma oranına göre sıcaklık güncellenmesi yapılarak döngü sonlandırılmakta ve sonraki tekrarlamaya geçilmektedir (Şekil 3.5).



Şekil 3.5. Benzetilmiş tavlama akış şeması

Özellik seçiminde, başlangıç pozisyonları rastgele permütasyon fonksiyonu ile belirlenmiştir. Bu fonksiyon girilen özelliklerden oluşan rastgele bir vektör döndürmektedir. Bulunan pozisyonlar uygunluk fonksiyonu ile değerlendirilerek bulunan en iyi çözüm atanmaktadır. En iyi maliyet değerlerini tutması için iterasyon adedi kadar eleman içeren bir dizi oluşturulmuştur. Başlangıç sıcaklığı ise 10 olarak girilmiştir.



Şekil 3.6. BT komşu oluşturma işlemlerinin oranı

Benzetilmiş tavlamanın toplam iterasyon adedi kadar çalışan ana döngüsünde, alt iterasyon sayısı kadar çalışan bir iç döngü daha bulunmaktadır. Yeni çözüm, komşu üretme fonksiyonu kullanılarak oluşturulmuştur. Bu fonksiyonda yer değiştirme, geri dönme ve katılma oranları sırasıyla 0,2, 0,5 ve 0,3 olarak uygulanmaktadır. Şekil 3.6'da gösterilen oranlar rulet çarkı seçimine sokularak çıkan sonuca göre bu işlemlerden biri başlangıçta belirlenen tura uygulanmaktadır. Yer değiştirme rastgele iki özelliği alarak dizideki pozisyonlarını değiştirmektedir. Geri dönme ise dizinin bir elemandan diğerine kadar olan kısmını alarak indeksleri kaydırmaktadır. Bu şekilde yeni tur belirlendikten sonra bu tura ait maliyet değeri hesaplanmıştır.

$$DELTA = (newsol.Cost - sol.Cost)/sol.Cost \tag{3.3}$$

$$P = \exp(-DELTA/T) \tag{3.4}$$

Yeni çözümün maliyeti, çözüm maliyetinden düşük ise bu çözüm güncel çözüm olarak atanmıştır. Eğer yeni çözüm daha iyi değil ise, benzetilmiş tavlama delta (Δ) değeri hesaplanmıştır. Delta komşu çözüm ile mevcut çözüm arasındaki farktır. Sonrasında Boltzmann dağılımı negatif deltanın sıcaklığa oranının üstel dağılımı şeklinde hesaplanmış ve kabul fonksiyonunun çıktısı elde edilmiştir. Bulunan değer daha iyiyse çözüm güncellemesi yapılmıştır. Böylelikle alt iterasyon tamamlanıp ana iterasyona ait döngüde en iyi maliyet değeri tutulmuş ve sıcaklık düşme oranına bağlı olarak sıcaklık güncellenmiştir.

3.2.4. Genetik Algoritma ile özellik seçimi

Bu çalışmada, özellik seçim yöntemlerinden yapay zekâ tabanlı genetik algoritma kullanılarak müzik veri setinde bulunan daha belirgin özellikler seçilmiştir. Başlangıç durumu ve değerlendirme kriteri belirlendikten sonra popülasyon sıralaması yapan ve bulunan en iyi çözümleri ve maliyet değerlerini tutan fonksiyonlar oluşturulmuştur. İterasyon başlangıcında ebeveyn seçimi yapılmıştır. Bu işlem yapılırken rulet çarkı seçimi kullanılmıştır. Çaprazlama adımında tek noktalı, iki noktalı ve üniform çaprazlama yapılarak yeni bireyler üretilmiştir. Yavruların uygunluğu, maliyet fonksiyonu ile değerlendirilerek bir sonraki adıma geçilmiştir.

Çizelge 3.3. Genetik algoritmada kullanılan parametreler

`nPop=50`	Popülasyon büyüklüğü
`pc=0.7`	Çaprazlama yüzdesi
`nc=2*round(pc*nPop/2)`	Yavru sayısı
`pm=0.3`	Mutasyon yüzdesi
`nm=round(pm*nPop`	Mutant sayısı
`mu=0.1`	Mutasyon oranı
`beta=8`	Seçim basıncı

Algoritmanın ilerleyen aşamasında seçilen özelliklere mutasyon uygulanmıştır. Bu işlemden geçen mutant özelliklerin uygunluğu test edilmiştir. Çaprazlama işlemine ve mutasyona uğrayan bireylerden oluşan birleşik popülasyon oluşturulmuştur. Oluşturulan bu nüfusa ait bireyler uygunluk değerlerine göre yeniden sıralandıktan sonra en kötü uygunluk değeri güncellenmiştir. Bulunan en iyi çözüm ve maliyet değerleri kaydedilmiştir.

Öncelikle veride bulunan değerler ve hedef değişkeni birlikte çekilmiştir. Çizelge 3.3'te bulunan parametreler tanımlandıktan sonra başlangıç aşamasına geçilmiştir. Bireylerin pozisyon ve maliyetlerini tutan boş bir yapı tanımlanmıştır. Bireylerden oluşan popülasyon boyutu kadar eleman tutan bir dizi oluşturulmuştur. Birey sayısı kadar çalışacak döngüde bitlerden oluşan genler kesikli tekdüze dağılım yapılarak elemanlara atanmıştır. Bireyler uygunluk fonksiyonu ile değerlendirilerek elde edilen değerler kaydedilmiştir. Popülasyondaki tüm bireyler uygunluklarına göre sıralanmıştır. En iyi çözüm kaydedilerek maliyet değerlerinin tutulacağı dizi yaratılmıştır. En kötü maliyetin bulunduğu değişken oluşturulmuştur.

Ana döngüde seçilim baskı sabitine göre kabul fonksiyonu değerleri alınmıştır. Çaprazlama işleminde kullanılacak boş matris oluşturulmuştur. Yavru sayısının yarısı kadar çalışacak olan çaprazlama döngüsünde ilk olarak ebeveynlerin indisleri kabul fonksiyonundan gelen değerlere göre rulet çarkına sokularak bulunmuştur. Bunun ardından, rulet çarkından gelen indislerde bulunan ebeveynler çaprazlama işlemi için seçilmiştir. Seçilen bireylere farklı türlerde çaprazlama yöntemleri uygulanmıştır. Tek noktalı, çift noktalı ve tek düze çaprazlama yöntemlerinden birinin seçilmesi için yine rulet çarkı seçimi kullanılmıştır. Bu yöntemlerin ruletteki oranları, tek noktalı için 0,1, çift noktalı için 0,2 ve tekdüze çaprazlama için 0,7 olarak belirlenmiştir. Çaprazlama yapıldıktan sonra oluşan yavruların uygunluğu değerlendirilerek döngü sonlandırılmıştır.

Mutasyon aşamasına gelindiğinde ise, öncelikle mutasyona uğrayacak popülasyonun bulunduğu mutant adedi kadar bireyden oluşan yapı kurulmuştur. Sonrasında ebeveyn seçimi rastgele şekilde yapılmıştır. Verideki değişken sayısı ile mutasyon oranının çarpımı üst tam sayıya yuvarlanarak mutasyona uğrayacak gen sayısı belirlenmiştir. Daha sonra, gen pozisyonlarından belirlenen mutant sayısı kadar gen indisi rastgele seçilmiştir. Ebeveynlerden gelen kromozomlar mutasyona uğrayacak bitlerin bulunduğu diziye gönderildikten sonra belirlenen indislerdeki gen değerleri 1'den çıkarılarak ters bite çevrilmiştir. Böylelikle mutasyon işlemi tamamlanmıştır. Ardından, mutantların uygunluğu maliyet fonksiyonu ile değerlendirilmiştir. Genetik algoritma operatörleri uygulandıktan sonra üretilen bireyler ile popülasyondaki tüm bireyler birleştirilmiştir. Popülasyon elemanları uygunluklarına göre sıralanmış ve en yüksek maliyet değeri tutulmuştur. Bulunan en iyi çözümler ve maliyet değerleri kaydedilmiştir.

3.2.5. Maliyet fonksiyonun belirlenmesinde Yapay Sinir Ağı kullanımı

Optimizasyon algoritmalarından gelen yeni çözüm adaylarının pozisyon değerleri listelenmektedir. Bu değerler yapay sinir ağlarına gönderilen verilerin içerisindeki indis numaralarını oluşturmaktadır. Bu veri seti içerisindeki değerler ve hedefler çok katmanlı yapay sinir ağlarına gönderilerek eğitim aşamasına alınır. Bu aşamada verilerin %70'i eğitim, %15'i test ve geri kalan %15'i ise doğrulama (validation) amacıyla bölünmüştür. Eğitim işlemi her seferde 3 kere tekrar edecek şekilde ayarlanmıştır.

Her çalışmanın sonunda elde edilen hata değerleri optimizasyon algoritmasına gönderilir. Yapay sinir ağlarından gelen hata değerleri denklem 3.5'te kullanılarak algoritmaların uygunluk fonksiyonu ile hesaplanır. Bu fonksiyondan gelen değerlerin ortalaması o anki aday çözümlerin uygunluğu olarak kaydedilir. Yeni çözümler bulunana kadar her iterasyonda bu değerler kontrol edilir. En iyi çözümler, yani en düşük hata değerine sahip olan öznitelikler seçilir. Bu şekilde seçim ve değerlendirme işlemi tamamlanır.

$$EE(r) = wTrain \times results.TrainData.E + wTest \times results.TestData.E \quad (3.5)$$

Bu formülde *wTrain* ve *wTest*, eğitim ve test verileri için ağırlık değerleridir. Tüm algoritmalarda bu değerler sırasıyla 0,8 ve 0,2 alınmıştır. Uygunluk fonksiyonunda eğitim ve

test işlemlerden gelen sonuçlar, bu ağırlıklar ile çarpılarak uygunluk değeri hesaplanmaktadır. *EE(r)* belirtilen tekrar sayısı sağlandıktan sonra elde edilen uygunluk (genel hata) değeridir.

Yapay sinir ağlarında genellikle çok katmanlı bir yapı vardır. Bu çalışmada meta-sezgisel algoritmalar ile entegre şekilde kullanılan yapay sinir ağında giriş ve çıkış katmanlarının yanında 10 gizli katman (hidden layer) bulunmaktadır. Eğitim yöntemi olarak ileri beslemeli ağlarda en hızlı öğrenme metodu olan Levenberg-Marquardt algoritması kullanılmıştır. Hedef ve çıkışlar arasında performans ölçümü karesel hata (mean squared error) fonksiyonu ile yapılmıştır.

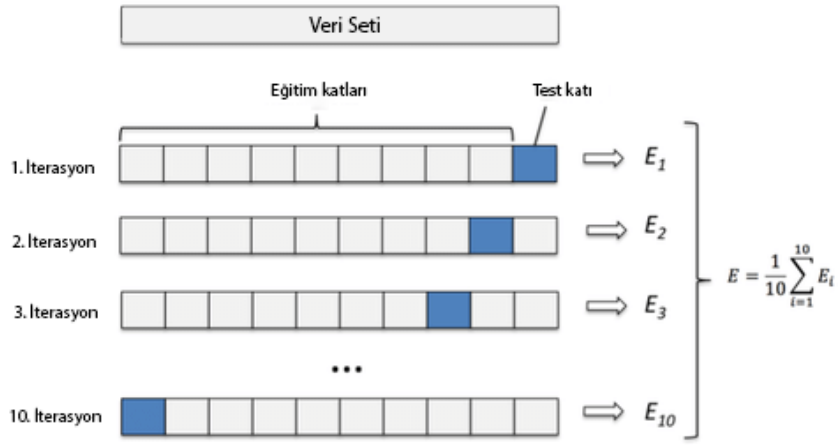
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f_i - y_i)^2 \tag{3.6}$$

Formülde, n veri noktalarının sayısı, f_i model tarafından döndürülen değer ve y_i ise bir i noktası için asıl değerdir.

3.2.6. Farklı sınıflandırıcılar ile seçim başarımı ölçümü

Bu çalışmada özellik seçiminden elde edilen sonuçların sınıf tahmin performansına etkisini ölçmek amacıyla farklı algoritmalarla sınıflandırma yapılmıştır. Metasezgisel yöntemler yapay sinir ağlarını kullanarak seçim yapacak şekilde tasarlanmıştır. Özellik seçme işlemi tamamlandıktan sonra bulunan özelliklerin diğer sınıflandırma yöntemleri ile sınıflama başarısı ölçülmüştür.

Sınıflandırma işlemi yapılırken 10 kat çapraz doğrulama (cross-validation) kullanılmıştır. Çapraz doğrulama, veri setini 9'u test ve 1'i eğitim için kullanılacak 10 rastgele alt kümeye böler. Bu süreç, tüm permütasyonlar eğitim ve test için kullanılana kadar 10 kez tekrarlanır. Şekil 3.7, k değerinin 10 olduğu k-kat çapraz doğrulamanın bir örneğini göstermektedir.



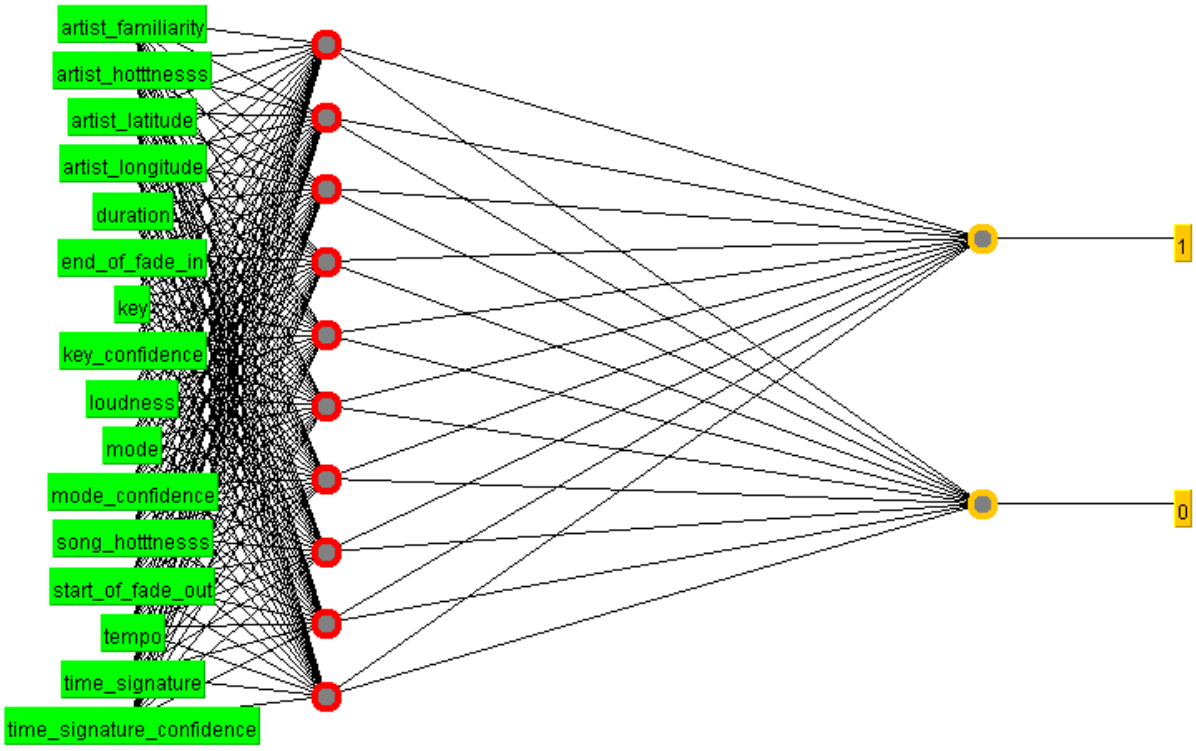
Şekil 3.7. 10 kat çapraz doğrulama örneği

- Naive Bayes algoritması, olasılık hesaplamaları yapan bir algoritmadır. Eğitim verilerini bir olasılık formülüne göre hesaplamaktadır. Farklı durumlar için bulunan olasılıklara göre test verisi üstünde sınıflandırma işlemleri yapmaktadır. Eğitilen verilerin boyutu fazla ise test aşamasında sınıfların başarılı biçimde tahmin edilme oranı da yüksek olmaktadır. Bu sınıflandırma yöntemi kullanılarak olasılık değerleri hesaplanırken, özelliklerin bağımsız olması önemli bir faktördür.

- Temeli C4.5 algoritmasına dayanan J48 karar ağacı Weka yazılımında bulunan bir algoritmadır. Bu model en yüksek bilgi kazancına sahip özellik belirlendikten sonra verilerin bölünmesi ile karara ulaşan düğüm yapısı oluşturmaktadır. J48 tüm özellikler için metrik bilgi kazancı hesaplamakta ve en yüksek kazanca sahip düğümü belirlemektedir. Bu durum oluşturulan karar ağacının sonuna kadar devam etmektedir.

- Destek Vektör Makinesi, regresyon analizi ve sınıflandırma için kullanılan istatistiksel öğrenme algoritmasıdır. Bu algoritmanın amacı farklı sınıflara ait destek vektörleri arasındaki mesafeyi maksimum hale getirmektir. Weka'da bulunan SMO (Sequential Minimal Optimization) ise bir destek vektör sınıflandırıcıyı eğitmek için John Platt'ın sıralı minimum optimizasyon algoritmasını kullanarak çalışan bir algoritmadır. Global olarak tüm eksik değerleri değiştirmektedir. Tüm nitelikleri varsayılan hale getirerek normalizasyon yapmaktadır. Bu sebeple, çıktıdaki katsayılar veri setinin orijinal haline değil normalize edilmiş versiyonuna göre bulunmaktadır.

- Weka makine öğrenme uygulamasında bulunan, k-en yakın komşuluk (kNN) algoritması türevlerinden biri olan IBk (Instance Based Learner), özellik uzayındaki en yakın eğitim örneklerine dayanarak test verilerini sınıflandıran, örüntü tanıma yöntemlerinden birisidir. Bu algoritma verilen *k* değeri kadar en yakın komşunun sınıfına göre sınıflandırma işlemi yapmaktadır. IBk algoritmasında bir vektörün sınıflandırılması, sınıfı bilinen vektörler kullanılarak yapılmaktadır. Bu çalışmada yapılan sınıflandırmalarda komşuluk belirten *k* değeri 3 alınmıştır. Komşu bulma işleminde lineer arama algoritması kullanılmıştır.



Şekil 3.8. 10 adet gizli katmana sahip bir Çok Katmanlı Algılayıcı

- Yapay sinir ağlarının bir sınıfı olan Çok Katmanlı Algılayıcı (MLP), derin, yapay bir sinir ağıdır. Birden fazla algılayıcıdan oluşur. Girdileri almak için bir giriş katmanı ve çıktıların sunulabilmesi için bir çıkış katmanı içermektedir. Bir ya da daha fazla gizli katman (Şekil 3.8) bulundurabilirler. MLP sinir ağının eğitimi, hata fonksiyonunun değerini minimum hale getirmek olarak tanımlanır. Weka yazılımında sınıflandırma yaparken, MLP sigmoid fonksiyonunu kullanmaktadır. İzlenebilir ve değiştirilebilir bir ağ yapısı sunmaktadır. Bu YSA, 0,3 öğrenme oranı ve 0,2 momentum katsayısına sahiptir.

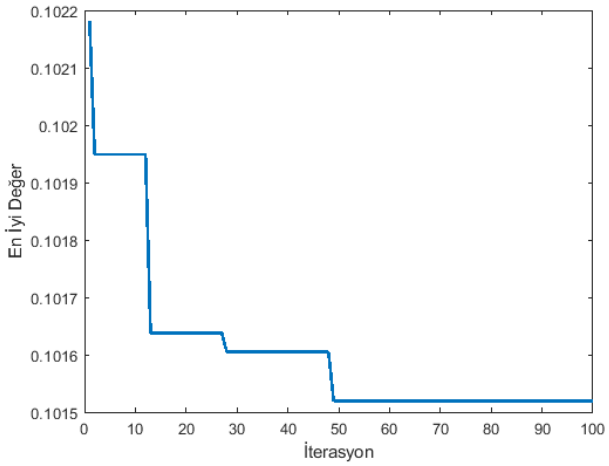
4. BULGULAR VE TARTIŞMA

Metasezgisel yöntemler kullanılarak yapılan çalışmalar kapsamında müzik veri setinde bulunan veriler içerisinden özellik seçimi yapılarak sınıflandırma başarımı artırılmıştır. Sınıflandırma işlemi için hangi özelliklerin daha önemli olduğu bulunmuştur. Veriyi anlamlı bir şekilde temsil edecek özelliklerin bulunduğu güncel veri seti oluşturulmuş ve veriler birçok sınıflandırıcı ile parça popülaritesine göre sınıflandırılmıştır.

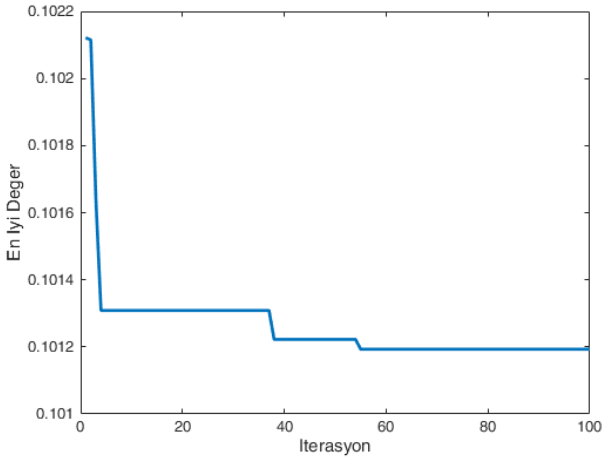
Çizelge 4.1. İterasyon sayısına göre algoritma karşılaştırma sonuçları

Yöntem	İterasyon	Öznitelik Kümesi	Min. Hata Değeri
KKA	20	[artist_hotttnesss, loudness, mode, mode_confidence]	0.10172
	50	[artist_hotttnesss, loudness, tempo, time_signature_confidence]	0.10153
	100	[artist_hotttnesss, loudness, tempo, start_of_fade_out]	0.10109
PSO	20	[artist_hotttnesss, loudness, tempo, key]	0.10122
	50	[artist_hotttnesss, artist_familiarity, tempo, start_of_fade_out]	0.10113
	100	[artist_hotttnesss, duration, tempo, end_of_fade_in]	0.10119
BT	20	[artist_hotttnesss, loudness, duration, time_signature]	0.10227
	50	[artist_hotttnesss, loudness, key, mode]	0.10193
	100	[artist_hotttnesss, loudness, key, mode_confidence]	0.10149
GA	20	[artist_hotttnesss]	0.10541
	50	[artist_hotttnesss]	0.10517
	100	[artist_hotttnesss]	0.10507

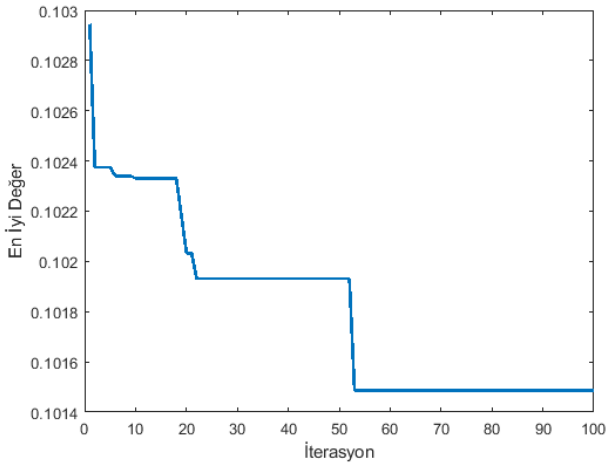
Sanatçı popülaritesini ifade eden *artist_hotttnesss* etiketinin, kullanılan algoritmaların tümünde önemli bir özellik olduğu görülmüştür. Sanatçı bilinirliğinin bir şarkının popüler olmasında önemli bir etken olduğu anlaşılmıştır. Bununla birlikte, parça temposu ve ses yüksekliğinin göstergesi olan *tempo* ve *loudness* özelliklerinin şarkı popülaritesinin belirlenmesinde önemli faktörler olduğu sonucuna varılmıştır. Yapılan 100 iterasyon sonunda en düşük hata değeri (Çizelge 4.1) karınca koloni algoritması ile elde edilmiştir.



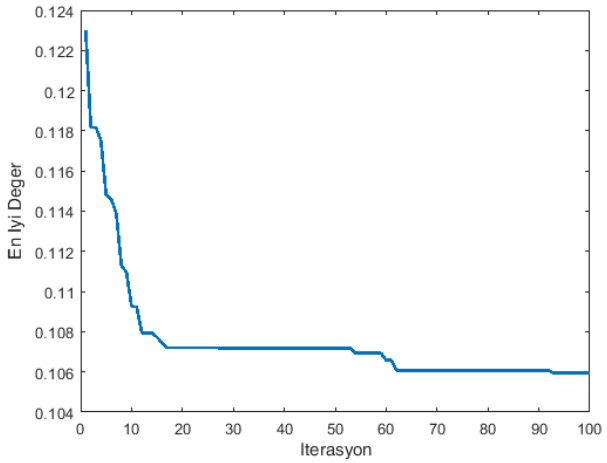
Şekil 4.1. a) Karınca koloni algoritması maliyet fonksiyonu grafiği



Şekil 4.1. b) Parçacık sürü optimizasyonu maliyet fonksiyonu grafiği



Şekil 4.1. c) Benzetilmiş tavlama maliyet fonksiyonu grafiği



Şekil 4.1. d) Genetik algoritma maliyet fonksiyonu grafiği

Çizelge 4.2. Sınıflandırma sonuçları

Sınıflandırıcı	Ham	KKA	PSO	BT	GA
IBk	84.53 %	84.96 %	85.04 %	84.83 %	85.48 %
Naive Bayes	84.77 %	87.86 %	87.75 %	87.97 %	88.00 %
MLP	87.74 %	88.08 %	88.02 %	88.04 %	88.07 %
J48	88.02 %	88.08 %	88.08 %	88.08 %	88.08 %
SMO	88.08 %	88.08 %	88.08 %	88.08 %	88.08 %

Özellik seçimi yapılmadan ve metasezgisel yöntemler kullanılarak özellik seçimi yapıldıktan sonra farklı sınıflandırıcılar ile sınıflama yapıldığında elde edilen sonuçlar karşılaştırılmıştır. Öncelikle ham veri üzerinde 5 farklı sınıflandırıcı ile yapılan sınıflandırmada en iyi başarım SMO algoritması ile elde edilmiştir. Çizelge 4.2'ye göre metasezgisel yöntemler ile özellik seçimi yapıldıktan sonra başarı oranının ham veri setine göre arttığı gözlenmiştir. Daha az özellik kullanılarak yapılan sınıflandırmalarda karar ağaçları, Naive Bayes, kNN, ve yapay sinir ağlarının başarı oranları verinin önceki haline göre artarken, destek vektör makineleri ile yapılan sınıflandırmada başarı oranında değişim olmamıştır. Bu sonuçlara göre, özellik seçimi ile elde edilen en yüksek başarı, bir karar ağacı algoritması olan J48 ile elde edilmiştir. Ham veri setine göre en yüksek başarım artışı ise, %3,23 oranındadır. Bu artış, genetik algoritma ile seçilen özellikler ve Naive Bayes sınıflandırıcısı kullanılarak elde edilmiştir. Çizelge 4.3'te bu sınıflandırıcı ile elde edilen hata oranları verilmiştir. Doğru sınıflandırılan örnek oranı azdan çoğa doğru sırasıyla, PSO, KK, BT ve GA ile elde edilmiştir.

Çizelge 4.3. Naive Bayes sınıflandırıcısı hata oranları

	KKA	PSO	BT	GA
TP Oranı	0,879	0,878	0,880	0,880
FP Oranı	0,862	0,858	0,867	0,871
Kesinlik	0,820	0,817	0,823	0,823
Duyarlılık	0,879	0,878	0,880	0,880
F-Ölçütü	0,829	0,829	0,828	0,827
MCC	0,064	0,064	0,058	0,050
ROC Alanı	0,593	0,584	0,588	0,581
PRC Alanı	0,822	0,818	0,820	0,817

Çizelge 4.4'deki sonuçlar incelendiğinde karınca koloni algoritması kullanılarak yapılan özellik seçiminin ardından k-en yakın komşuluk sınıflandırması ile toplamda 8497 tane örneğin doğru sınıflandırıldığı görülmektedir. Bu oran verilerin %84,96'sına denk gelmektedir. Bunlardan 86 tanesi popüler ve 8411 tanesi popüler olmayan kategorisinde bulunmuştur. Geriye kalan 1504 örnek ise hatalı sınıflandırılmıştır. Parçacık sürü optimizasyonu ve Naive Bayes sınıflandırıcısı kullanılarak yapılan sınıflandırma işleminde ise Çizelge 4.5'te görüldüğü üzere 32 adet TP ve 8744 adet TN değerine ulaşılmıştır. Böylelikle verilerin %87,75'inin doğru sınıflandırılmış olduğu görülmektedir.

Çizelge 4.4. KKA - kNN karışıklık matrisi

86	1106
398	8411

Çizelge 4.5. PSO - Navie Bayes karışıklık matrisi

32	1160
65	8744

5. SONUÇ VE ÖNERİLER

5.1. Sonuçlar

Meta-sezgisel yöntemler, sınıflandırma ve özellik seçimi gibi makine öğrenmesine ait farklı alanlarda başarıyla kullanılmaktadır. Bu tezde dört farklı meta-sezgisel yöntemden yararlanılmıştır. Karınca koloni algoritması, parçacık sürü optimizasyonu, benzetilmiş tavlama ve genetik algoritma yöntemleri özellik seçimi amacıyla kullanılmıştır. Bu yöntemlerin, performans farklılıkları karşılaştırılmış ve veri seti içerisinden en anlamlı olan özellikleri seçen modeller oluşturulmuştur. Bu şekilde sınıflandırma başarımının artırılması sağlanmıştır.

Özellik seçimi işleminde meta-sezgisel algoritmalar ve yapay sinir ağlarından yararlanılmıştır. Öncelikle veri setinde bulunan nümerik özellikler algoritmalar aracılığıyla seçilmiştir. Seçilen özelliklerin uygunluğunun değerlendirilmesi amacıyla ise yapay sinir ağlarından faydalanılmıştır. Daha sonra, anlamlı özelliklerin bulunduğu güncellenmiş veri setinin diğer sınıflandırıcılar ile başarımı karşılaştırılmalı olarak test edilmiştir. Naive Bayes, k-en yakın komşuluk, destek vektör makineleri ve karar ağaçları gibi birçok farklı sınıflandırma yöntemi bu amaçla kullanılmıştır. Sonuç olarak hangi sınıflandırıcıların işlenen veri setinde daha iyi performans verdiği gözlemlenmiştir.

5.2. Öneriler

Gelecekteki çalışmalarda, halihazırda uygulanan algoritmaların hız performansları test edilebilir. Farklı parametre değerleri ile süre ve maliyet alanlarında iyileştirmeler yapılabileceği mümkün görünmektedir. Ayrıca, diğer meta-sezgisel algoritmaların özellik seçimi başarısını ölçmek amacıyla yeni çalışmalar yapılabileceği düşünülmektedir. Sezgisel optimizasyon yöntemlerinin hibrit versiyonlarının da bu amaçla kullanılması önerilmektedir.

Özellik seçimi yapıldıktan sonra elde edilen özelliklerin kullanılan sınıflandırma yöntemleri ile tahmin başarımını artırmak için sınıflandırıcıların probleme uygun şekilde özelleştirilmesiyle daha başarılı sonuçlar elde edilebilir. Sınıflandırma probleminin çözümü

için farklı sınıflandırma yöntemlerinin de denenmesinin faydalı olacağı tahmin edilmektedir. Kullanılan veriden daha büyük boyutlu veri setleri üzerinde analizler yapılmak suretiyle makine öğrenmesine ait eğitim ve test aşamaları güçlendirilebilir ve daha yüksek başarı oranları elde edilebilir.

KAYNAKLAR DİZİNİ

Arat, B., 2018, Model Performans Değerlendirme Ölçütleri-1, https://berkarat.com, erişim tarihi: 24.10.2019.

Aggarwal, C., 2014, Data Classification Algorithms and Applications, Chapman and Hall/CRC.

Aydın, İ., 2015, Karınca Koloni Algoritması BMÜ-579 Meta Sezgisel Yöntemler, http://web.firat.edu.tr/iaydin/bmu579/bmu_579_bolum6.pdf, erişim tarihi: 09.07.2019.

Baig, M., Aslam, N., Shum, H., 2019, Filtering Techniques for Channel Selection in Motor Imagery EEG Applications: A Survey, Artificial Intelligence Review.

Bertin-Mahieux, T., Ellis, D., Whitman, B., Lamere, P., 2011, The Million Song Dataset, ISMIR 2011: Proceedings of the 12th International Society for Music Information Retrieval Conference.

Bilgin, M., 2017, Gerçek Veri Setlerinde Klasik Makine Öğrenmesi Yöntemlerinin Performans Analizi, 19. Akademik Bilişim Konferansı - AB 2017.

Budak, H., 2018, Özellik Seçim Yöntemleri ve Yeni Bir Yaklaşım, Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 22, Özel Sayı, 21-31.

Büyükoflaz, N., Öztürk, A., 2018, Early Autism Diagnosis of Children with Machine Learning Algorithms, 2018 26th Signal Processing and Communications Applications Conference.

Canayaz, M., Demir, M., 2017, Feature Selection With The Whale Optimization Algorithm And Artificial Neural Network, 2017 International Artificial Intelligence and Data Processing Symposium (IDAP).

Çetişli, B., 2006, Öznitelik Seçiminde Dilsel Kuvvetli Sinir Bulanık Sınıflayıcı Kullanımı, Eskişehir Osmangazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 19,2.

Dabbabi, K., Hajji, S., Cherif, A., 2017, Integration Of Evolutionary Computation Algorithms And New AUTO-TLBO Technique In The Speaker Clustering Stage For Speaker Diarization Of Broadcast News, EURASIP Journal on Audio, Speech, and Music Processing.

Dash, S., Patra, B., 2017, Feature Selection Algorithms for Classification and Clustering in Bioinformatics, IGI Global, p.111.

Dikbayır, S., 2017, MetaSezgisel Algoritmalar, https://biryazilimciningunlugu.wordpress.com/2017/05/16/metasezgisel-algoritmalar, erişim tarihi: 12.08.2019.

KAYNAKLAR DİZİNİ (devam)

Doğan, B., Korurek, M., 2012, ECG Beat Clustering Using Fuzzy C-means Algorithm And Particle Swarm Optimization, 20th Signal Processing and Communications Applications Conference (SIU).

Faragardi, H., Shojaee, R., Keshtkar, M., Tabani, H., 2013, Optimal Task Allocation for Maximizing Reliability in Distributed Real-Time Systems. 2013 IEEE/ACIS 12th International Conference on Computer and Information Science, 513-519.

Filiz, D., Tanrıöver, Ö., 2020, An Exploration of Machine Learning Methods for Biometric Identification Based on Keystroke Dynamics, IGI Global, 258-269.

Hosny, M., 2012, Vehicle Routing with Pickup and Delivery: Heuristic and Meta-heuristic Solution Algorithms, LAP Lambert Academic Publishing.

Huang, D., Han, K., Hussain, A., 2016, Intelligent Computing Methodologies: 12th International Conference, Proceedings, Part III.

Humeau, J., Liefooghe, A., Talbi, E., Verel, S., 2013, ParadisEO-MO: From Fitness Landscape Analysis to Efficient Local Search Algorithms. Journal of Heuristics, 19.

Keskintürk, T., Söyler H., 2006, Global Karınca Kolonisi Optimizasyonu, Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 21,4, 689-698.

Liang, Z., Lou, L., Liu, J., 2019, Fitness Functions for Forward Selection: Application in Seven Data Sets, International Journal of Quantitative Structure-Property Relationships, 4 70-87.

Liu, H., Yu, L., 2005, Toward Integrating Feature Selection Algorithm for Classification and Clustering. IEEE Transaction on Knowledge and Data Engineering 17(4), 491-502.

Maimon, O., Rokach, L., 2005, Data Mining and Knowledge Discovery Handbook, Springer Nature, p.372.

Nasreldin, M., Ma, S., Dailey, E., Dang, P., 2018, Song Popularity Predictor, Towards Data Science.

Öztemel, E., 2012, Yapay Sinir Ağları, Papatya Yayıncılık, s.19.

Patel, S., 2017, Chapter 0 : What Is Machine Learning?, https://medium.com/machine-learning-101/chapter-0-what-is-machine-learning-ad136361c618, erişim tarihi: 25.09.2019.

Siddique, N., Adeli, H., 2013, Evolutionary Computing, Computational Intelligence: Synergies of Fuzzy Logic, Neural Networks and Evolutionary Computing, pp. 183-237.

KAYNAKLAR DİZİNİ (devam)

Şeker, S., 2013, Sınıflandırma (Classification), http://bilgisayarkavramlari.sadievrenseker.com/2013/03/31/siniflandirma-classification, erişim tarihi: 06.05.2019.

Talbi, E., 2009, Metaheuristics: From Design to Implementation, John Wiley & Sons, p.240.

Willems, F., 1982, The Feedback Capacity Region of A Class of Discrete Memoryless Multiple Access Channels (Corresp.), IEEE Transactions on Information Theory, 28,1, 93-95.

Yarpiz Project, 2016, Feature Selection using Metaheuristics and EAs, http://yarpiz.com/306/ypml122-evolutionaryfeature-selection, erişim tarihi: 25.02.2019.