

Gen Ekspresyonları ile Hastalık Arasındaki İlişkiyi Pi Algoritmasını Kullanarak Belirleme

Determining the Relationship Between Gene Expressions and Disease Using the Pi Algorithm

Murat DEMİR^{*1} 

¹Software Engineering, Muş Alparslan University, Muş, Türkiye

(m.demir@alparslan.edu.tr)

Received: May 01, 2026

Accepted: Jun.08, 2026

Published: Jun.09, 2026

Özetçe— Multidisipliner çalışma, günümüz bilim dünyasında önemi giderek artan bir konu olmuştur. Özellikle bilgisayar teknolojisindeki gelişmeler, bu bilim dalı ile her disiplini beraber kullanmayı artık bir zorunluluk haline getirmiştir. Bilgisayar mühendisliği akademik çalışmalarında, optimizasyon problemlerine yeni bir bakış açısı getiren meta sezgisel yöntemler; özellikle problemin parametre sayısının ve hacminin büyük olduğu durumlarda ve klasik yollarla problemin çözümünün güç ve çok zaman alması halinde devreye girmektedir. Bu çalışmada literatüre yeni önerilen ve Pi sayısının üretilmesini taklit ederek çalışan yeni bir meta sezgisel yöntem olan Pi Algoritması kullanılmıştır.

Algoritma, hastalık ve gen arasındaki ilişkiyi temsil eden ve Kaggle veri tabanından elde edilmiş, Gene Expression Analysis and Disease Relationship isimli veri seti üzerine uygulanmıştır. 1000 adet veri içeren veri içeren veri setinin 970 adet verisi eğitim için kullanılmıştır. Sonuç olarak; modelin ne kadar doğru çalıştığını gösteren doğruluk (accuracy) için en yüksek 0.96, aynı kümedekilerin benzerliklerini ölçen kesinlik (precision) için en yüksek 1, benzer olanların hepsinin kapsanmasını ölçen duyarlılık (recall) için en yüksek 0.909, kesinlik ve duyarlılık arasındaki dengeyi ölçen F1 skor için en yüksek 0.952 performans değerlerine ulaşılmıştır.

Anahtar Kelimeler : Meta Sezgisel Yöntemler, Pi Algoritması, Kümeleme.

Abstract— Multidisciplinary study has become an increasingly important topic in today's scientific world. In particular, advancements in computer technology have made it a necessity to integrate this field with all other disciplines. In computer engineering academic studies, heuristic methods, which offer a new perspective on optimization problems, come into play especially when the number and size of parameters in the problem are large, and when solving the problem using classical methods is difficult and time-consuming. In this study, the Pi Algorithm, a new metaheuristic method newly proposed in the literature and working by imitating the generation of the number Pi, was used.

The algorithm was applied to the Gene Expression Analysis and Disease Relationship dataset, obtained from the Kaggle database, which represents the relationship between disease and genes. 970 data points from the 1000-point dataset were used for training. As a result; the highest performance values achieved were: accuracy (showing how correctly the model works) at 0.96, precision (measuring the similarities of those within the same cluster) at 1, sensitivity (measuring the coverage of all similar ones) at 0.909, and the F1 score (measuring the balance between accuracy and sensitivity) at 0.952.

Keywords : Meta Heuristic Methods, Pi Algorithm, Clustering.

1. Giriş

Hızla gelişen teknoloji birçok bilim dalında baş döndürücü ilerlemelere ön ayak olmaktadır. Bilgisayar teknolojisinin bu kadar hızlı gelişim göstermesi, beraberinde bu teknolojiyi her geçen gün tüm bilim dalları ile beraber çözüm üretmede mümkün kılmaktadır. Yıllar öncesine bakıldığında manuel olarak yapılan süreçlerin hemen hemen tümü artık bilgisayar teknolojisi sayesinde, otomatik olarak değerlendirilmekte ve işlenmektedir.

Olayların ve kavramların birbirine daha çok bağlı olduğu bir dünyada, özellikle de toplumu ilgilendiren zorlukların ya da çözülmesi gereken problemlerin karmaşıklığı, tek disiplinli çalışma kavramının daha ilerisinde bir çözüm gerektiğini açıkça göstermektedir. Geleneksel çözüm yaklaşımları, çok yönlü sorunları ele almada çoğu zaman yetersiz kalmaktadır. Farklı alanlardan yöntemleri ve bilgileri birleştiren çok disiplinli yaklaşımlar problemlerin çözümünde daha kapsamlı ve daha anlaşılır çözümler sunabilmektedir. Her disiplinin kendine ait güçlü yönlerinden ve bakış açısından faydalanarak, yenilikçi ve etkili çözümler sunabilmektedirler (Khan, 2024).

Multidisipliner çalışma, farklı disiplinleri ilgilendiren ortak bir problemi çözmek için farklı disiplinlerin işbirliğini gerektirir. Multidisiplinerlik, her alanın sınırlarını koruyarak, ortak bir hedefe bağımsız olarak katkıda bulunmalarına imkân sağlar. Bu yaklaşım, tek disiplinin problemi çözmek için yetersiz kalabileceği karmaşık çalışma alanlarında özellikle etkilidir (Klein, 2010).

Bu bakış açısıyla multidisipliner çalışma; ekonomi, sağlık, eğlence, üretim teknolojisi, astronomi, malzeme bilimi, sosyal ve toplumsal bilimler vb. tüm bilim dallarında, bilgisayar teknolojisinde geline seviyeden faydalanmaktadır. Çünkü geline noktada elde edilen verinin boyutu ve hacmi giderek artmaktadır ve böylesine büyük verileri anlamlandırmak ve işleyebilmek manuel olarak yapılmaktan öteye gitmiştir.

Bilgisayar bilimleri alanında özellikle yapay zeka ve makine öğrenmesinin geldiği nokta baş döndürücü bir seviyeye ulaşmıştır. Çok devasa veriler bu teknolojiler yardımıyla analiz edilebilmektedir. Bu teknolojik gelişmelere paralel olarak hayatın her alanında dijital ortamlarda çok büyük boyutlu veriler yapay zekâ teknolojilerinin hammaddesini meydana getirmektedir (Nergiz, 2022).

2. Optimizasyon

Optimizasyon kavramı, bir sistem, süreç ya da bir tasarımı mümkün olan en iyi duruma kavuşturmayı hedefleyen matematiksel yöntemleri kullanmaktır. Bu süreç genelde, amaçlanan hedeflere kavuşmak adına; sistem, süreç veya tasarımın iyileştirilmesi çalışmalarını barındırır (Mei & Wang, 2021).

Optimizasyon, çoğu bilim dalında kullanılan bir kavramdır. İşletmelerde üretimin planlanması, envanter yönetimi ve lojistik gibi birçok alanda kullanılabildiği gibi, mühendislik alanında tasarımın ve süreçlerin iyileştirilmesi gibi süreçlerde de yaygın olarak kullanılmaktadır (Beauregard et al., 2008).

Gerçek dünya problemlerinin optimizasyonu için en çok başvurulanan yöntemler sayısal yöntemler olmakla beraber; bazı açılardan sayısal yöntemlerle bu optimizasyonları gerçekleştirmek çok zor olabilmektedir. Örneğin, başlangıç noktalarının problemin çözümü için çok kritik olması, gradyan bilgisinin gerekmesi gibi durumlar, global çözüm aramayı çok zorlaştırmakta ve çözümü kararsız hale getirebilmektedir. Doğrusal olmayan karmaşık kısıtlamalar ve amaç fonksiyonları içeren, çoklu karar değişkenleri barındıran sistemler geleneksel sayısal yöntemlerle çözülmesi gittikçe zorlaşan bir yapıya kavuşmuştur (Zaloğlu et al., 2023).

2.1. Meta sezgisel Yöntemler

Meta sezgisel yöntemler, geleneksel yöntemlerle çözülemeyen ya da çözümü çok zaman alabilen, çok boyutlu optimizasyon problemlerinde kullanılan yöntemlerdir. Bu yöntemler; doğadaki olaylar ya da sistemlerden ilham almaktadırlar. Problemlerin çözümünde global en iyi çözümü bulabilmek için geniş bir çözüm uzayını keşfetmektedirler (Almufti et al., 2023).

Meta sezgisel algoritmalar, küresel bir çözüme yaklaşmak için deterministik olmayan yöntemler kullanır ve bir sistemin gerçek davranışını temel alan matematiksel bir model kurmak amacıyla kullanılabilir. Birçok farklı yapıda ve özellikte meta sezgisel algoritma mevcuttur (Kuvat&Kuvat, 2025).

İyi bilinen meta sezgisel yöntemlere örnek olarak: insan beynini (McCulloch & Pitts, 1943), kuş ve balık sürülerini (Kennedy et al., 1995), insan bağışıklık sistemini (Alataş & Akın, 2004), arıları (Karaboğa, 2005), ateş böceklerini (Yang, 2008), yer çekimini (Rashedi et al., 2009), yarasaları (Yang, 2010), gri kurtları (Mirjalili, S. et al., 2014), balinaları (Mirjalili & Lewis, 2016) vb. bir çok canlıyı ve sistemi örnek alan yöntemler verilebilir.

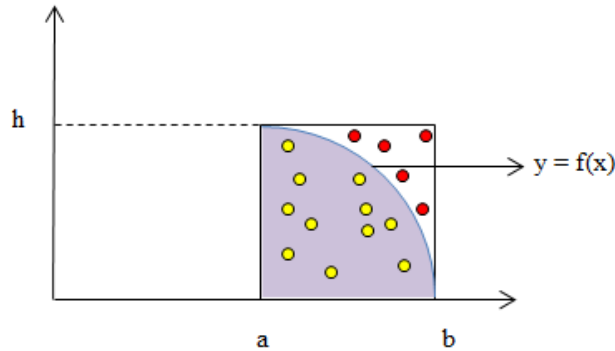
Meta sezgisel yöntemlerin hepsi en iyi olan çözüme benzemeye çalışmak amacıyla çözümler üretirler. En iyi çözümün etrafında çözümler üreterek optimum çözümü elde etme amacındadırlar. Bu amaçla her meta sezgisel yöntemin kendine ait matematiksel bir alt yapısı vardır. Çözüm popülasyonlarını üretirken bu matematiksel alt yapıyı kullanırlar.

2.1.1. Pi Sayısı ve Pi Algoritması

2.1.1.1. Pi Sayısının Hesaplanması

Pi sayısının ilk hesaplanmaya çalışıldığı dönemlerde rasyonel bir sayı olduğu düşünülse de, Lambert 1761'de yaptığı çalışmada (Lambert, 1761) Pi sayısının bir irrasyonel sayı olduğunu göstermiştir. Bu ispat sonrasında Pi sayısının hesaplanması için birçok çalışma yapılmıştır. Bu çalışmalardan biri de monte-carlo metodu olmuştur. Monte-Carlo metodu, Nicholas Constantine Metropolis ve Stanislaw Ulam (Metropolis & Ulam, 1949) tarafından ortaya atılmış ve daha sonra Metropolis ve diğerleri (Metropolis vd., 1953) tarafından daha da geliştirilmiştir.

Monte-Carlo metodu, entegrasyon üzerine kurulu bir metottur. Rasgele sayıları kullanarak hesaplamasını gerçekleştirir. Literatürdeki öteki metotlar düzgün dağılıma sahip bir entegrasyon yapma üzerine kurgulanırken; Monte-Carlo metodu rasgelelik üzerine bir entegrasyon hesaplama metodu seçmiştir. İntegral hesaplaması yapılırken, rasgele kombinasyonlar oluşturularak hesaplama gerçekleştirilir (Gültekin & Asyalı, 2007). Şekil 1.'de Monte-Carlo yönteminin bir $f(x)$ fonksiyonu için $[a,b]$ aralığında integralinin hesaplanması temsil edilmiştir.



Şekil 1. Monte-Carlo integralinin görsel temsili

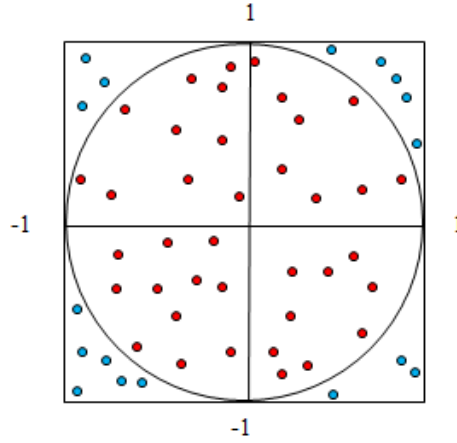
$f(x)$ fonksiyonu bu dikdörtgeni iki farklı bölgeye ayırır. Birinci bölge, hesaplanmak istenen integralin bulunduğu fonksiyonun altındaki alandır. İkinci bölge ise fonksiyonun üst kısmındaki alandır. Rasgele oluşturulacak olan noktalar $[a,0]$, $[b,0]$, $[a,h]$ ve $[b,h]$ koordinatları ile belirlenen dikdörtgenin içerisine denk gelecek şekilde belirlenir. Noktaların bir kısmı çeyrek dairenin içine bir kısmı ise çeyrek dairenin dışına denk gelecektir. Fonksiyonun altındaki nokta sayısının (P_x) toplam nokta sayısına (P_t) oranı, bu fonksiyonun altındaki alanın (A_x) dikdörtgenin alanına (A_t) oranının yaklaşık bir ifadesini verecektir. Denklem (1) bu ilişkiyi ifade etmektedir.

$$\frac{P_x}{P_t} = \frac{A_x}{A_t} \quad (1)$$

Entegrasyon hesaplaması için iki rastgele sayı üretildiğini düşünecek olursak; üretilen ilk sayı, $[a, b]$ aralığındaki x , diğeri ise $[0, h]$ aralığındaki y değeri içindir. $f(x)$ fonksiyonundan elde edilen sayı ile y değeri karşılaştırıldığında y sayısının $[0, h]$ aralığında olup olmadığı anlaşılır. Eğer $y - f(x)$ değeri sıfır ya da negatif bir sayı ise, rasgele üretilen değerler taralı alana düştüğü anlaşılır. Bu şekilde birçok (x, y) çifti üretilir ve dairesel bir alana uygulanacak olursa, Denklem (2)'deki ifade elde edilir (Metropolis & Ulam, 1949).

$$z = f(x, y) = x^2 + y^2 < 1 \quad (2)$$

1 değeri, birim çemberin yarıçapıdır. $z(x, y)$ koordinatlarına sahip karenin içine rastgele noktalar yerleştirirsek, bunlardan bazıları çemberin içinde, bazıları ise dışında kalacaktır. Şekil 2 bu durumu temsil eden şekli ifade etmektedir.



Şekil 2. Birim çember üzerinde Monte-Carlo noktalarının üretiminin temsili

Birim çemberin alanı π , karenin alanı ise 4 birimdir. Bu iki değeri birbirine oranlayacak olursak, Denklem (3)'deki formülü elde ederiz.

$$\frac{\text{Probability of being inside the circle}}{\text{Probability of being inside the square}} = \frac{\pi}{4} \quad (3)$$

Atışların çemberin içine düşme olasılığını, karenin içine düşme olasılığıyla karşılaştırıp elde edilen değeri 4 ile çarparsak, π sayısının yaklaşık değerini elde ederiz. Nokta sayısı arttıkça, istenen değere daha yakın bir sayı elde edilir.

2.1.1.2. Pi Algoritması

Pi algoritması, Pi sayısının Monte-Carlo metodu ile hesaplanması üzerine dayalı bir Meta sezgisel algoritmadır. Literatüre Demir (Demir, 2024) tarafından önerilmiştir. Pi algoritması yardımıyla, Monte-Carlo yöntemi kullanılarak r , (x,y) , z ve Pi katsayıları hesaplaması modellenmiştir.

Şekil 2.'deki birim çemberin yarıçapı olan r değeri, Denklem (4)'te ifade edildiği şekliyle, Pi algoritmasında hesaplanmaya çalışılan parametre değerlerinin minimum ve maksimum değerinin yarısı olarak kabul edilir. Bir çemberin çap değeri, çember üzerindeki birbirlerinden en uzak noktalardır. Bu yaklaşımla; r değeri, veri kümesindeki bir öznitelik/parametre için maksimum ve minimum değerler arasındaki fark olarak hesaplanır. Bu değer yarısı ise yarıçap olarak ifade edilmiştir.

$$r = \frac{\max(\text{attribute}_i) - \min(\text{attribute}_i)}{2} \quad (4)$$

Denklem 2'de bahsi geçen z değerleri hesaplanırken, eldeki veri setinin aralıkları dikkate alınarak normalize edilmiş x ve y değerlerinden yola çıkarak hesaplama yapılmıştır. Böylece yeni üretilmiş olan z değerlerinin veri seti ile bağlantılı bir şekilde istenilen aralığın dışına çıkması önlenmiştir. Denklem (5) ve (6) bu durumu ifade etmektedir.

$$x = \min(\text{attribute}_i) + (\max(\text{attribute}_i) - \min(\text{attribute}_i)) * rand \quad (5)$$

$$y = \min(\text{attribute}_i) + (\max(\text{attribute}_i) - \min(\text{attribute}_i)) * rand \quad (6)$$

z sayısı üretilen rasgele çözüm değerinin çemberin içinde kalıp/kalmadığını ölçen değerdir. r 'den küçükse, çemberin içinde olduğu kabul edilir. Bu çalışmada Denklem(4)'te belirtilen r değerinden küçük olması halinde değer istenilen aralıkta üretilmiş olur. Çünkü r sayısı, öznitelik değerlerinin birim çemberi olarak kabul edilen çemberin yarıçapıdır. Yoksa z değeri çemberin dışında kabul edilir. Bu durum Denklem (7)'de ifade edilmiştir.

$$z = x^2 + y^2 \leq \frac{\max(\text{attribute}_i) - \min(\text{attribute}_i)}{2} \quad (7)$$

Bu çalışmada mesafe ölçüsü olarak Öklid mesafesi ölçüsü (Pandit & Gupta, 2011) kullanılmıştır. Mesafe, üretilen her bir x ve y değerinin maksimum değerden farkıdır (Demir, 2024). Çünkü x ve y birim çemberde merkezden (0,0) olan uzaklıkları temsil etmektedir. Önerilen algoritmada ise, veri setindeki maksimum değerden olan uzaklığı temsil etmektedir. Bu yaklaşım her öznitelik için ayrı ayrı uygulanır. Yani her özniteliğin kendine ait bir maksimum değeri ve hesaplanacak bir Pi katsayısı olacaktır. Sonuçta elde edilen bir Pi katsayılar matrisidir.

Algoritma 1, önerilen Pi algoritması ışığında Monte-Carlo metodu ile hesaplanan Pi katsayılarının Pseudo kodunu göstermektedir (Demir, 2024).

Algoritma 1: Monte-Carlo metodu ile yeni Pi katsayılarının üretilmesi

1. Procedure Pi_coefficient
 2. while sayaç<=adım sayısı
 3. $x \leftarrow [\min_attribute_i, \max_attribute_i]$ aralığında rasgele bir sayı üret
 4. $y \leftarrow [\min_attribute_i, \max_attribute_i]$ aralığında rasgele bir sayı üret
 5. $z \leftarrow \sqrt{(x - \max(\text{attribute}_i))^2 + (y - \max(\text{attribute}_i))^2}$
 6. eğer $z \leq \frac{\max(\text{attribute}_i) - \min(\text{attribute}_i)}{2}$
 7. aralıktaki_değer_sayısı $\leftarrow$ aralıktaki_değer_sayısı ++
 8. end eğer
 9. sayaç++
 10. end while
 11. Pi_coefficient $\leftarrow 4 * (\text{aralıktaki_değer_sayısı} / \text{sayaç})$
 12. end Procedure
-

Algoritma 1'deki Pi katsayıları kullanılarak yeni önerilen Pi algoritması Algoritma 2'deki halini almaktadır (Demir, 2024).

Algoritma 2: Pi Algoritması Pseudo kodu

1. Procedure Pi_algorithm
 2. Çözüm için veri seti ile aynı sütun sayısında rasgele bir popülasyon üret
 3. Global en iyi çözüm için veri seti ile aynı sütun sayısında tek satırlık başlangıç değerlerini rasgele ata
 4. İlk popülasyonun içerisinde uygunluk fonksiyonuna göre en iyi başlangıç çözümünü ata
 5. Bir x ve y değer çifti üret
 6. z değerini hesapla $z \leftarrow \sqrt{(x - \max(\text{attribute}_i))^2 + (y - \max(\text{attribute}_j))^2}$
 7. z değerinin Denklem(4)'te verilen değerden küçük olması/olmaması durumuna göre çemberin içine denk gelen değerleri say ve ona göre her öz niteliğe ait Pi katsayıları hesapla
 8. While sayaç < adım sayısı
 9. Her aday çözümün gerçek veri seti ile arasındaki fark (uygunluk) değerlerini hesapla
 10. Uygunluk değerlerine göre popülasyondaki çözümleri sırala ve sıra indekslerini belirle
 11. En küçük uygunluk değerine sahip indeksi çözümü seç
 12. Lokal en iyi çözüm $\leftarrow$ En iyi çözüme sahip değer
 13. Eğer Lokal en iyi çözüm < Global en iyi çözüm
 14. Global en iyi $\leftarrow$ Lokal en iyi çözüm
 15. end Eğer
 16. Yeni çözüm popülasyonunu hesapla $\text{pop}_{i,t+1} = \text{pop}_{i,t}(i,j) + (\text{best} - \text{pop}_{i,t}) * \text{Pi}_{\text{coefficient}} * \text{rand}$
 17. End while
 18. En iyi çözüm değerini yazdır
-

3. Veri Kümeleme

Yapay zekanın bir alt çalışma alanı olan makine öğrenmesi, bilgisayarlara algoritmalar aracılığıyla öğrenme yeteneği kazandıran disiplindir. İnsanlara özgü olan şeyleri yapabilen modeller oluşturmak için algoritmaları veriler üzerinde eğiten yapay zekânın (AI) bir dalı olarak da düşünülebilir. Bilgisayarların eldeki verilerden öğrenmesini kolaylaştıran bilimdir. Yapay zekâ, makineleri insan gibi davranabilen hale getirmek için kullanılan farklı stratejiler ve teknikler için kullanılan genel bir terimdir. Makine öğrenmesi ise, makinelerin deneyimlerden yola çıkarak, öğrenme ve uyum sağlama yeteneğine sahip olması gerektiği fikrine dayanmaktadır (Sadiku vd. , 2025; Yalamanchili, 2023).

Makine öğrenmesi yöntemleri temelde 3 yöntem şeklinde sınıflandırılır: Eğitici öğrenme, eğitici öğrenme ve takviyeli öğrenme.

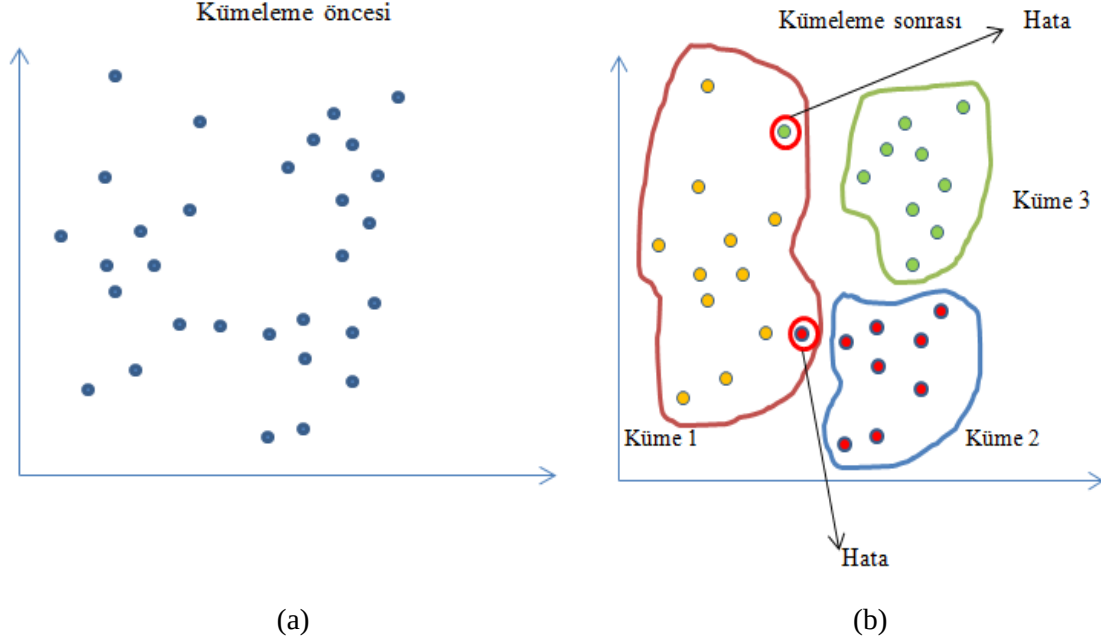
Eğitici öğrenme, etiketli veriler kullanılarak modellerin eğitilmesini dayanır (Bishop, 2006). Öğrenme olayı, girdi-çıktı çiftlerine dayanır ve model, eğitim süreci boyunca girdileri çıktılara eşlemeyi öğrenir.

Eğitici öğrenme, etiketlenmemiş verilerle çalışır (Han vd. , 2011). Bu öğrenme yönteminde, model çıktılar hakkında önceden bilgi sahibi olmadan kalıpları, grupları veya yapıları belirlemeye çalışır. Denetimsiz öğrenme yöntemleri genellikle mesafe ölçütlerine, benzerlik ölçütlerine ve olasılık dağılımlarına dayanır.

Takviyeli öğrenme, ödül ve ceza sistemi ile karar vermeyi amaçlar (Sutton & Barto, 2018). Öğrenme, ödüller veya cezaların geri bildirimlerini kullanarak en uygun kalıpları öğrenmek için ajanlar içerir. Öğrenme olayı bu yolla gerçekleşen; keşif, sömürü ve ödül mekanizmaları ile gerçekleşir (Yalamanchili, 2023).

Kümeleme ise bir eğitici öğrenme yöntemidir. Kümeleme, bir veriyi özniteliklerine göre farklı kümelere gruplandırmayı amaçlayan bir kavramdır. Kümeleme teknikleri, tutarlı bir gruplandırma

sağlamak için veri noktaları arasındaki mevcut benzerlikleri belirler (Pileggi, 2025). Şekil 3. kümeleme yönteminin çalışma prensibini temsil eden genel bir yaklaşımı temsil etmektedir.



Şekil 3. Kümeleme yönteminin grafiksel gösterimi

Şekil 3.1.'de gösterildiği üzere; Şekil 3.1. (a), kümeleme öncesi verinin bir bütün halinde bulunduğu durumdur. Şekil 3.1. (b) ise, kümeleme işleminden sonraki durumdur. 2 nokta kümeleme işleminde hatalı hesaplanmıştır. Küme 1'de bulunan 2 değer, aslında biri Küme 2'de, biri de Küme 3'de olması gereken noktalardır. Bu durumda bu değerler hesaplamada hatalı olarak kaydedilir.

Kümeleme işleminde performansı hesaplamak için kullanılan metrikler vardır. Bu metrikler kullanılarak kümelemenin; doğruluğu, geçerliliği ve eksiksizliği ölçülür. Bunun için karışıklık matrisi adı verilen bir matris kullanılır. Tablo 1. Karışıklık matrisinin temsili göstermektedir (Yeşildal, 2022).

Tablo 1. Karışıklık matrisi

	Gerçek pozitif	Gerçek negatif
Tahmin edilen pozitif	TP (True Pozitif)	FP (False Pozitif)
Tahmin edilen neegatif	FN (False Negatif)	TN (True Negatif)

Performans metrikleri ise, Accuracy, Precision, Recall ve F1 skor değerleri kullanılır. Accuracy modelin ne kadar doğru çalıştığını, Precision sonuçların ne kadar geçerli olabileceğini, Recall sonuçların ne kadar eksiksiz olduğunu, F1 skor ise geçerlilik ve eksiksizlik arasındaki mevcut dengeyi ifade eder. Denklem (8), (9), (10) ve (11) bu metriklerin formüllerini göstermektedir (Aslanyürek & Mesut, 2022).

$$Doğruluk = \frac{TP+TN}{TP+TN+FP+FN} \quad (8)$$

$$Kesinlik = \frac{TP}{TP+FP} \quad (9)$$

$$Duyarlılık = \frac{TP}{TP+FN} \quad (10)$$

$$F1 \text{ skor} = 2 * \frac{Kesinlik * Duyarlılık}{Kesinlik + Duyarlılık} \quad (11)$$

4. Deneyisel Sonuçlar

Bu çalışmada, Kaggle'dan (Kaggle, 2026) ulaşılan Gene Expression Analysis and Disease Relationship isimli veri seti üzerinde çalışılmıştır. Gen ekspresyon profillerini ve ilgili klinik özellikleri detaylandıran 1000 hipotetik örnek mevcuttur.

Veri setinin 8 özniteliği vardır. Bunlardan biri ID numarasıdır. Bu öznenin çıkışa sayısal olarak hiçbir etkisi yoktur. Bu nedenle eğitim verisinin içerisinde çıkarılmıştır. Geri kalan öznitelikler aşağıdaki gibidir:

Yaş: Hastanın yaşı, Yaş ve gen ekspresyon profilleri arasındaki potansiyel korelasyonları araştırmak için kullanılabilir.

Cinsiyet (Kategorik): Hastanın cinsiyeti (0: Kadın, 1: Erkek). Cinsiyetin gen ekspresyonu veya hastalık durumu üzerindeki etkileri analiz edilebilir.

Gen_X_Ekspresyonu (Sayısal): Belirli bir genin, Gen X göreceli ekspresyon seviyesi. Bu, hastalık ilerlemesinde veya tedavi yanıtında rol oynayabilecek varsayımsal bir geni temsil eder.

Gen_Y_Ekspresyonu (Sayısal): Başka bir belirli genin, Gen Y göreceli ekspresyon seviyesi. Gen X ile birlikte veya ondan bağımsız olarak incelenebilir.

Sigara Durumu (Kategorik): Hastanın sigara içme durumu (0: Sigara kullanmayan, 1: Eski sigara kullanan, 2: Halen sigara kullanan). Çevresel faktörlerin gen ekspresyonu ve hastalık üzerindeki etkisi değerlendirilebilir.

Hastalık Durumu (Kategorik): Hastanın hedef hastalıkla ilgili durumu (0: Sağlıklı, 1: A Hastalığı, 2: B Hastalığı). Bu, tahmin modelleriniz için birincil hedef değişken olarak kullanılabilir.

Tedaviye Yanıt (Kategorik/Sayısal): Uygulanan tedaviye verilen yanıt derecesi (0: Yanıt Yok, 1: Kısmi Yanıt, 2: Tam Yanıt). Gen ekspresyon profillerinin tedavi başarısını tahmin etmedeki rolü incelenebilir.

Bu çalışma bir kümeleme çalışması olduğundan TreatmentStatus (Hastalık durumu) öznenin veri setinden çıkarılmıştır. Çünkü kümeleme yönteminde çıkış değeri kullanılmadan gruplama yapılır. Bunun için Algoritma 2'nin 9. Adımında belirtilen uygunluk hesaplamasının değerleri kullanılır.

Veri seti hem sayısal hem de kategorik veri içermektedir. Kategorik veriler: Gender (2. Öznenlik), SmokingStatus (5. Öznenlik), TreatmentResponse (6. Öznenlik) ve DiseaseStatus (çıkış öznenin) öznenlikleridir. Bunlar veri setinde 0-1, 0-2, 0-2 ve 0-2 şeklinde kategorize edilmiştir.

Eğitim için ayrılan veri, modelin eğitimi için kullanılır. Test verisi ise modelin doğru çalışıp/çalışmadığını görmek için kullanılır. Veri setinde toplamda 1000 adet örnek vardır. Hastalığın derecesine göre (0, 1 ve 2), her gruptan 10; toplamda 30 tane örnek test için ayrılmıştır. 970 adet örnek eğitim için kullanılmıştır. Orijinal veri setinde tedaviye yanıt derecesi 1 ve 2 olan guruplardan mevcut örneklerin sayısı çok azdır. Bu nedenle test için çok örnek ayrılması durumunda bu gurupların eğitim verisi az kalacaktır. Bu gurupları öğrenmede problem çıkabilecektir. Bu, eğitim için dezavantajlı bir durum olacağından; test için her gruptan 10 tane örnek yeterli görülmüştür.

Pi algoritmasının Monte-Carlo adımıyla iterasyon sayısı 1000 olarak seçilmiştir. Yani, entegrasyon ile Pi katsayıları belirlenirken 1000 kez hesaplama yapılmıştır. Çözüm değerlerinin Pi algoritmasına göre yeniden üretimi için de 1000 kez tekrar sağlanmıştır. Popülasyon en iyi çözümü elde edinceye kadar 1000 kez üretilmiştir. Her adımda yeniden en iyi seçimi gerçekleştirilmiştir.

Tablo 2. Modelleme ile elde edilen ve algoritmanın 1000 adım çalışmasından sonra elde edilen Pi katsayılarını göstermektedir.

Tablo 2. Pi Algoritması ile elde edilen Pi katsayıları

Pi_katsayısı 1	Pi_katsayısı 2	Pi_katsayısı 3	Pi_katsayısı 4	Pi_katsayısı 5	Pi_katsayısı 6
0,824	0,808	0,760	0,744	0,796	0,804

Tablo 2’deki katsayılar kullanılarak elde edilen best matris ise Tablo 3.’deki gibidir.

Tablo 3. Pi algoritması ile elde edilen Best matris değerleri

best_1	best_2	best_3	best_4	best_5	best_6
0,9156	0,1399	0,9954	0,9547	0,1712	0,6831

Test değerleri için, best matris değerleri kullanılarak elde edilen küme aralıkları ise Tablo 4.’deki gibidir.

Tablo 4. Pi algoritmasının elde ettiği küme aralıkları

Küme 1 (Sağlıklı)	Küme 2 (A hastalığı)	Küme 2 (B hasatlığı)
21,782	54,800	74,566
26,892	56,020	76,556
33,172	59,410	77,886
33,893	61,350	80,956
49,373	63,140	84,106
60,532	63,790	87,666
69,273	64,300	91,556
88,723	66,630	92,776
89,833	67,760	93,996
92,772	68,070	95,216

Tablo 4. dikkatlice incelenecek olursa, sağlıklı kümeye ait olan 5 adet test verisinin aralık değerinin, A hastalığı ve B hastalığının küme aralığı ile karıştığı görülmektedir. Bu da 5 adet verinin algoritma tarafından yanlış tespit edildiğini geri kalan 25 örneğin ise kümesinin doğru tespit edildiği görülmektedir. Tablo 4.’deki verilerden yola çıkarak karışıklık matrisi oluşturulduğunda Tablo 5.’deki karışıklık matrisi elde edilir.

Tablo 5. Her gurup için elde edilen karışıklık matrisi değerleri

	Sağlıklı	Hastalık A	Hastalık B
Sağlıklı	5	2	3
Hastalık A	0	10	0
Hastalık B	0	0	10

Karışıklık matrisin değerlerinden yola çıkarak her kümeye ait doğruluk (accuracy), kesinlik (precision), duyarlılık (recall) ve F1 skor değerleri Tablo 6.'daki gibi elde edilir.

Tablo 6. Performans metriklerinin sonuçları

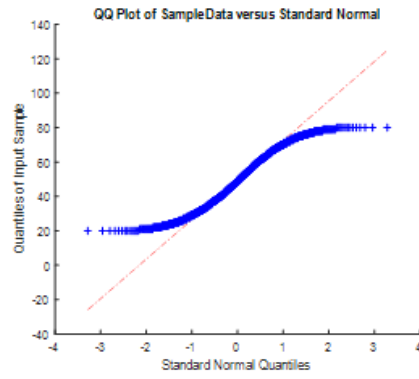
	Doğruluk	Kesinlik	Duyarlılık	F1
Küme 1	0,83	0.5	1	0,909
Küme 2	0,96	1	0,909	0,952
Küme 3	0,9	1	0,769	0,869
Ortalama	0,897	0,833	0,892	0,91

Tablo 6.'daki değerler incelendiğinde ortalama doğruluk, kesinlik, duyarlılık ve F1 skor değerlerinin başarılı sonuçlar verdiği görülmektedir. Burada incelenmesi gereken bir husus vardır. Demir'in (Demir, 2024;Demir, 2025) 2 çalışmasında da belirttiği üzere; özniteliklerin dağılımındaki Z skor değerleri ve öznitelikler arasındaki korelasyon değerleri de performansı etkilemektedir. Bu veri setine ait Z skorları ifade eden Q-Q plot grafiği Şekil 4'te verilmiştir.

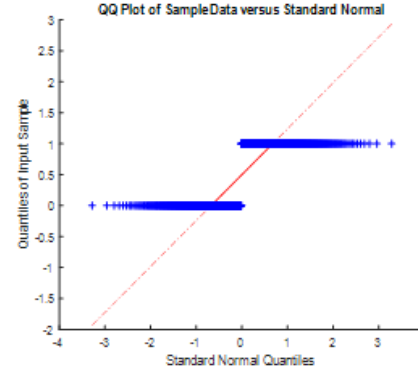
Şekil 6 .dikkatlice incelenecek olursa; öznitelik 2 dışındaki özniteliklerin dağılımı kırmızı kesikli çizginin yakınlarında bir dağılıma sahip değildir. Bu da örneklemin normal dağılımdan gelip gelmediğini gösteren önemli bir işarettir. Bu da bize göstermektedir ki; veri setimizin öznitelik 2 değerleri hariç diğer değerleri normal dağılıma yakın değerlere sahiptir. Buda kümelemede o özneliği ayırt edici bir özellik yapmaktan uzaklaştırmaktadır. Her ne kadar 0.8-0.9 aralığında çıkan doğruluk, kesinlik, duyarlılık ve F1 değerleri performans için iyi sonuçlar olsa da; Pi algoritmasının daha yüksek performans değerlerine ulaşmada özniteliklerin ayırt edici bir dağılıma sahip olmaması yani normal dağılım göstermesi etkili olmuştur. Çünkü kümeleme özniteliklerin yakınlığına göre grupları belirleyen bir strateji uygular. Özniteliklerin dağılımı da normal dağılım etrafında olursa bu performansı etkileyecektir.

Ayrıca veri setindeki özniteliklerin korelasyonlarına da bakarsak bunu görebiliriz. Şekil 5. korelasyon matrisini göstermektedir.

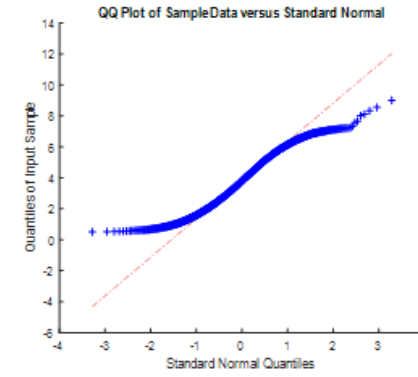
Şekil 5 dikkatlice incelenecek olursa; 1, 3 ve 4 numaralı özniteliklerin diğer özniteliklerle güçlü bir pozitif korelasyona sahip olduğu durumlar vardır. 2,5 ve 6. öznitelikler ise zayıf korelasyonlara sahiptir. Şekil 4 ile Şekil 5 beraber değerlendirilecek olursa; Q-Q plot grafiğinde ayırt edici dağılıma sahip olan en belirgin öznitelik 2 numaralı özniteliktir. Çünkü veriler kırmızı eğim çizgisinden uzak alanda dağılım göstermiştir. Fakat bu özneliğin ise diğer öznitelikler ile korelasyonu zayıftır. Bu sebeple algoritma özellikle küme 1 için kesinlik değeri (precision) çok düşük bir değerde kalmıştır. Bu da gösteriyor ki; verideki dağılım, özniteliklerin korelasyon değerleri performans metriklerinin değerlerinde etkili olmaktadır.



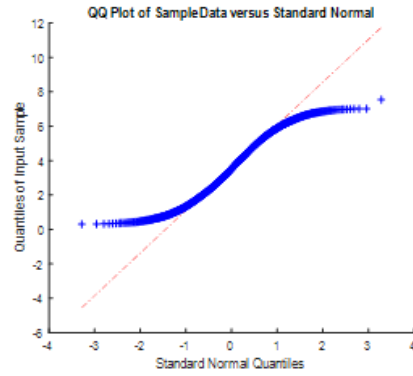
Öznitelik 1



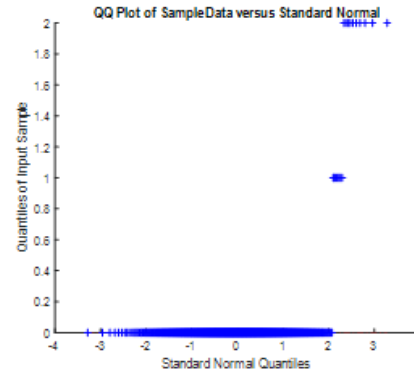
Öznitelik 2



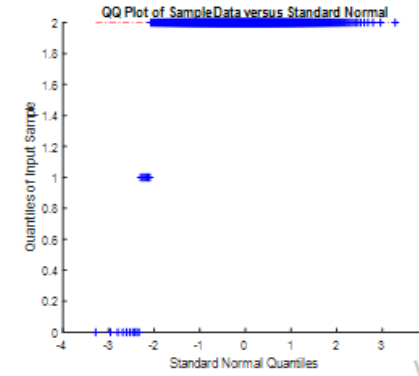
Öznitelik 3



Öznitelik 4



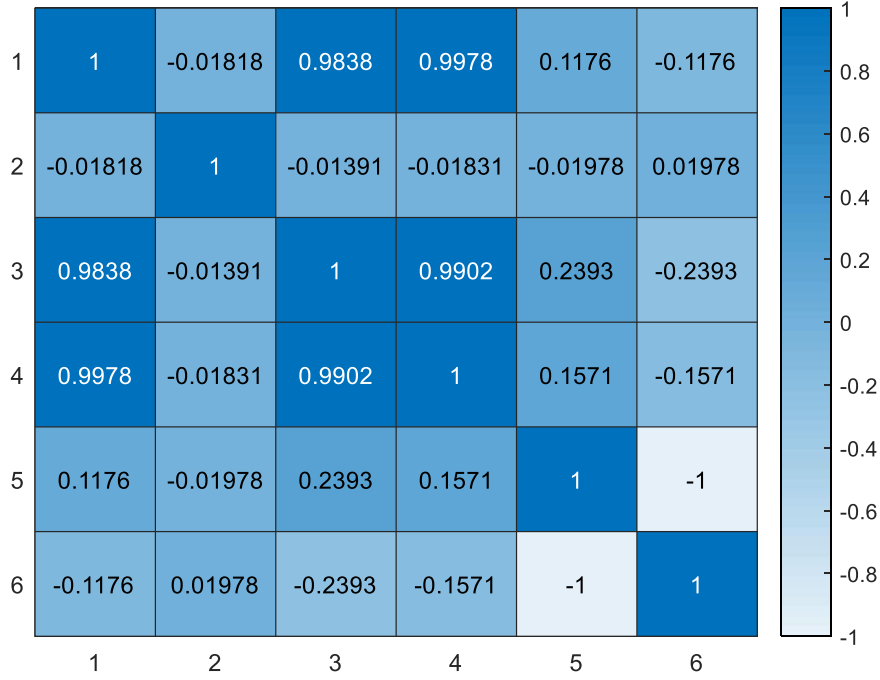
Öznitelik 5



Öznitelik 6

Win
Wind

Şekil 4. Veri seti Q-Q plot grafiği



Şekil 5. Korelasyon matrisi

Sonuç

Literatüre yeni önerilen Pi algoritması, hem fazla parametre kullanmaması hem de matematiksel alt yapısının sadeliği yönüyle avantajlı bir meta sezgisel yöntemdir. Bu çalışmada literatüre yeni sunulan algoritma, kümeleme işlemi gerçekleştirmek için kullanılmıştır ve anlamlı sonuçlar elde edilmiştir. Gen ekspresyonları ile hastalık arasındaki ilişkiyi belirleme açısından başarılı bir şekilde çalışmıştır.

Performans metrikleri; doğruluk, kesinlik, duyarlılık ve F1 için ortalama 0.9 civarı rakamlara ulaşılmıştır. Bu da küme için başarılı sayılacak değerlerdir. Literatüre yeni önerilmesi ve henüz geliştirme aşamasında olmasına rağmen, başarılı bir şekilde kümeleme işlemini gerçekleştirmiştir.

Özellikle büyük veri seti için algoritmanın nasıl performans gösterdiği ileriki çalışmalarda araştırma konusu olmaya açıktır. Pi katsayılarını üretirken yüksek durdurma kriterine sahip döngülerde ya da başlangıç popülasyonu üretilirken yüksek boyutlu popülasyonlar üretildiğinde hesaplama zamanı açısından çok fazla olmasa da bir miktar gecikme yaşanabilmektedir. Bunları tolere etmek için bir takım çalışmalar yapılabilir.

Ayrıca algoritmanın kemik hesaplama parametrelerinden olan; r ve z değerinin daha optimize karşılıkları kümeleme için denenebilir. Yeni popülasyon üretilirken kullanılan esas formülde güncellemeler denenebilir.

Kaynaklar (References)

Alataş, B, Akın E. (2004) Yapay Zekada Yeni Bir Alan: Yapay Bağışıklık Sistemleri, YA/EM'2004-Yöneylem Araştırması/Endüstri Müh.-XXIV Ulusal Kongresi

- Almufti SM, Shaban AA, Ali ZA, Ali RI and Fuente J.D (2023). Overview of metaheuristic algorithms. *Polaris Glob. J. Sch. Res. Trends* 2: 10-32.
- Beauregard Y, Thomson V & Bhuiyan N (2008). Lean engineering logistics: Load leveling of design jobs with capacity considerations. *Can. Aeronaut. Space J.* Vol. 54(2): 19–30.
- Bishop CM. (2006) Pattern Recognition and Machine Learning, *Springer*
- Demir M. (2024) A New Meta-Heuristic Method: Pi Algorithm and An Its Application on Data Clustering. *Preprints* <https://doi.org/10.20944/preprints202409.0478.v1>
- Demir M. (2025) Artificial Afterimage Algorithm: A New Bio-Inspired Metaheuristic Algorithm and Its Clustering Application. *Applied Sciences* 15, 1359, <https://doi.org/10.3390/app15031359>
- Kaggle (2026) Gene Expression Analysis and Disease Relationship, <https://www.kaggle.com/datasets/ylmzasel/gene-expression-analysis-and-disease-relationship> Accessed 02 April 2026
- Gültekin AT & Asyalı MH. (2007) Pi Sayısının Monte-Carlo Metodu ve Gregory/Leibniz Formülüyle Hesaplanması. *Yaşar Üniversitesi E-Dergisi* 2(7):751-760 , <https://doi:10.19168/jyu.42351>
- Han J. vd. (2011). Data Mining: Concepts and Techniques . Elsevier.
- Johann Heinrich Lambert. (1761) Mémoire sur quelques propriétés remarquables des quantités transcendentes circulaires et logarithmiques. *Histoire de l'Académie Royale des Sciences et Belles-Lettres (Berlin)*, XVII,265–322.
- Karaboğa D. (2005) An idea based on honey bee swarm for numerical optimization (Vol. 200). *Technical report-tr06* Erciyes university, engineering faculty, computer engineering department
- Kennedy J & Eberhart R. (1995) Particle swarm optimization. In *Proceed-ings of IEEE international conference on neural networks*, 4:1942–1948. Perth, Australia.
- Khan A. (2024) Integrating Perspectives: The Role of Multidisciplinary Approaches in Solving Complex Problem. *Kash Journal of Multidisciplinary Research* 1(8):400-410.
- Klein JT (2010) Creating interdisciplinary campus cultures: A model for strength and sustainability. John Wiley & Sons.
- Kuvat G & Kuvat Ö. (2025) Metasezgisel Algoritma Seçiminde Kriter Önceliklerinin AHP Yöntemiyle Belirlenmesi. *Türk Mühendislik Araştırma ve Eğitimi Dergisi*, 4(2): 197-205.
- Matthew NO Sadiku , Samuel A Ajayi & Janet O Sadiku. (2025) Machine Learning: An Overview. *International Journal of Trend in Scientific Research and Development*. ISSN: 2456-6470, 9(5):1121-1131.
- Mcculloch, WS & Piitts W. (1943) A Logical Calculus of The Ideas Immanent in Nervous Activity. *Bulletin of Mothemnticnl Biology* 52(1/2): 99-115.
- Mei L-B, & Wang Q. (2021) Structural optimization in civil engineering: A literature review. *Buildings* 11(2): 66, <https://doi.org/10.3390/buildings11020066>
- Metropolis N & Ulam S. (1949) The Monte Carlo Method. *Journal of the American Statistical Association* 44: 335-341, <https://doi.org/10.2307/2280232>

- Metropolis N, Rosenbluth, AW, Rosenbluth MN, Teller AH & Teller E. (1953) Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics* 21: 1087- 1092, <http://dx.doi.org/10.1063/1.1699114>
- Mirjalili S, Mirjalili SM, & Lewis A. (2014) Grey wolf optimizer. *Advances in Engineering Software* 69: 46–61.
- Mirjalili S. & Lewis A. (2016) The Whale Optimization Algorithm. *Advances in Engineering Software* 95:51-67.
- Nergiz M. (2022) İşbirlikçi Yapay Zeka Konsepti: Federe Öğrenmeye Genel Bir Bakış. *Dicle University Journal of Engineering* 13(2): 279-286, <https://orcid.org/0000-0002-0867-5518>.
- Pandit S. & Gupta S. (2011) A comparative Study on Distance Measuring Approaches for Clustering. *International Journal of Research in Computer Science* eISSN 2249-8265 2(1): 29-31.
- Pileggi SF. (2025) Clustering with Uncertainty: A Literature Review to Address a Cross-Domain Perspective. *Informatics* 12(2), 38, <https://doi.org/10.3390/informatics12020038>
- Rashedi E, Nezamabadi-pour H & Sarazdi S. (2009) A gravitational search algorithm. *Information Science* 179: 2232-2248.
- Yalamanchili CM. (2023) A Comprehensive Study of Machine Learning Models - Types, Examples, and Use Cases. *International Journal For Multidisciplinary Research* 5(1), [doi:10.36948/ijfmr.2023.v05i01.41033](https://doi.org/10.36948/ijfmr.2023.v05i01.41033)
- Yang XS (2008) Nature-Inspired Metaheuristic Algorithms. Frome: LuniverPress. ISBN 1-905986-10-6.
- Yang XS. (2010). A New Metaheuristic Bat-Inspired Algorithm. In J. González, D. Pelta, C. Cruz, G. Terrazas & N. Krasnogor (Eds.), Nature Inspired Cooperative Strategies for Optimization (NICSO 2010) 284:65-74, Springer Berlin Heidelberg.
- Zaloğlu M, Fidan Ş & Erkan E. (2023) Meta-Heuristik Optimizasyon Algoritmalarının Sistem Tanımlama Problemine Uygulanması ve Performans Karşılaştırması. 2nd International Conference on Engineering, Natural and Social Sciences, April 4-6, Konya, Türkiye.