En Yakın k Komşuluk Algoritmasında Örneklere Bağlı Dinamik k Seçimi

Faruk BULUT[1] M. Fatih AMASYALI[2]

[1,2] Bilgisayar Mühendisliği Bölümü
Elektrik-Elektronik Fakültesi
Yıldız Teknik Üniversitesi, Davutpaşa, İSTANBUL

Email: f0110303@std.yildiz.edu.tr mfatih@ce.yildiz.edu.tr

Özet

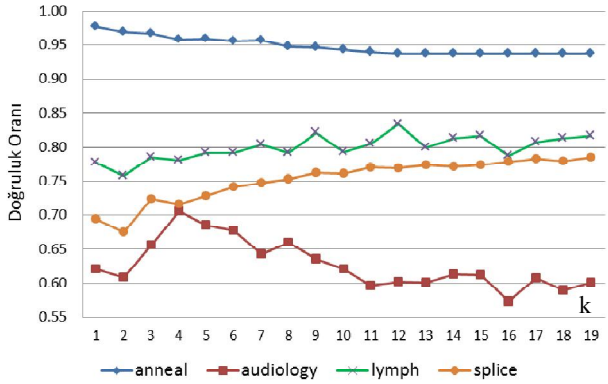
k en yakın komşuluk algoritması(k-NN)içinen uygun k parametresi, kullanıcı tarafından genellikle deneme-yanılma yöntemiyle seçilir. Bununla birlikte, bir veri setinde her bir test örneği için aynı k parametresinin kullanılması genel sınıflandırma başarısını olumsuz etkileyebilir. Çalışmamızda her bir test örneği için en uygun k parametresini kümeleme yöntemiyle bulan ve bu sayede genel sınıflandırma başarısını artıran bir yöntem üzerinde çalışılmışve başarılı sonuçlar elde edilmiştir.

1. Giriş

*k*en yakın komşuluk algoritması(*k*-NN), sabit bir *k* değeri ile kullanılmaktadır. Literatürde en uygun *k* değerinin deneme yanılma yöntemi ile bulunduğu belirtilmektedir [1]. Sabit *k* değerinin genel sınıflandırma başarısı üzerindeki olumsuz etkileri bu çalışmada incelenecek ve bir takım çözümler önerilecektir.

Şekil-1'de görüldüğü üzere 4 adet UCI veri seti [2] üzerinde *k*-NN sınıflandırıcısısın *k* parametresi 1'den 20'ye kadar sırayla denenmiş ve doğruluk (*accuracy*) oranları elde edilmiştir. Görüldüğü üzere *k* parametresinin artırılmasıyla sınıflandırma başarısı bazılarında artmış, bazılarında azalmış, bazılarında da bir değişme olmamıştır. Bu durum *k*-NN algoritmasının *k* parametresine olan hassasiyeti göstermektedir.

Her bir örneği doğru sınıflandırmak amacıyla faklı bir *k* değerine gereksinim duyulduğu Tablo-1'de görülmektedir. *audiology* veri setinin test kısmında bulunan bazı örneklerin geçerleme işleminde sınıflandırıcının farklı *k* değerleriyle bu test örneğini doğru sınıflandırıp sınıflandırmadığı gözlemlenmeye çalışılmıştır. 1 değeri sınıflandırıcının ilgili örneğin etiketini doğru tahmin ettiğini; 0 ise yanlış tahmin ettiğini göstermektedir. Bu tabloya göre *k* değeri 1 alındığında 8 örnekten sadece 3 tanesi doğru bilinebilirken; *k* değeri 4 alındığında 6 tanesi doru bilinebilmektedir.



Şekil 1. k-NN'de k parametresinin performansa etkisi

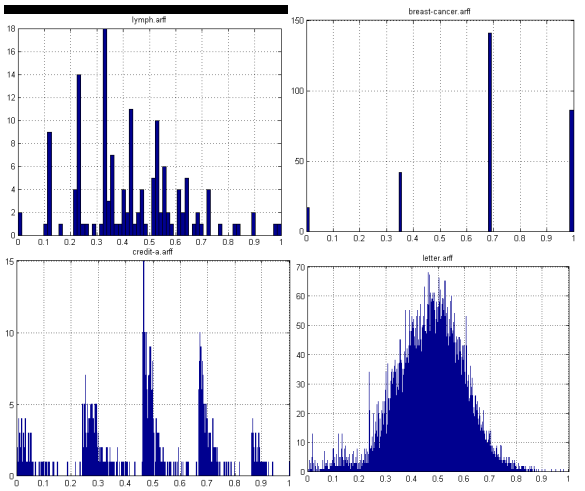
Tablo-1'de görüldüğü üzere bazı test örnekleri için *k* değeri artırıldığında, azaltıldığında ya da belirli aralıklarda alındığında doğru tahmin yapılmaktadır. Bu durum bize genel sınıflandırma başarısının artırmak amacıyla her test örneği için sabit bir *k* değeri yerine uygun bir *k* değerinin seçilmesi gerektiğini göstermektedir.

Tablo 1. k değerinin başarıya etkisi

	k-NN için *k* parametresi									
	1	2	3	4	5	6	7	8	9	10
1.örnek	0	0	0	0	0	0	0	1	1	1
2.örnek	1	1	1	1	1	1	1	1	1	1
3.örnek	0	0	0	0	0	0	0	0	0	0
4.örnek	0	0	1	1	1	1	1	1	1	1
5.örnek	0	0	0	1	1	1	0	0	0	1
6.örnek	1	1	1	1	1	1	1	1	0	0
7.örnek	1	1	1	1	1	0	1	1	1	1
8.örnek	0	0	0	1	0	0	0	0	0	0
Doğru tahmin sayısı	3	3	4	6	5	4	4	5	4	5

Şekil-2'denormalize edilmiş UCI veri setlerinden sırayla *lymph*, *breast-cancer*, *credit-a* ve *letter*'ın orijin noktalarına göre tüm noktaların uzaklıklarının histogramları gözükmektedir ve veri setlerinin kendi

uzaylarındaki yayılımları hakkında bir fikir vermektedir. Orijindeki bir test noktasının sınıflandırılmasında histogram grafiğine bakacak olursak en uygun *k* değeri *lymph* veri setinde 2; *breast-cancer*'da 15 alınmalıdır. Çünkü orijindeki noktaya aynı uzaklıkta olan ve aynı yörünge üzerinde duran sırasıyla 2 ve 15 tane nokta kümeleri vardır. *letter* ve *credit-a* veri setinde ise en uygun k değeri için 1'den başlayarak deneme yanılma yöntemi uygulanabilir. Bu durum bize herhangi bir test örneğinin etrafındaki noktaların uzaklıklarına bağlı olarak *k*-NN için uygun bir k parametresinin bulunabileceğini göstermektedir.



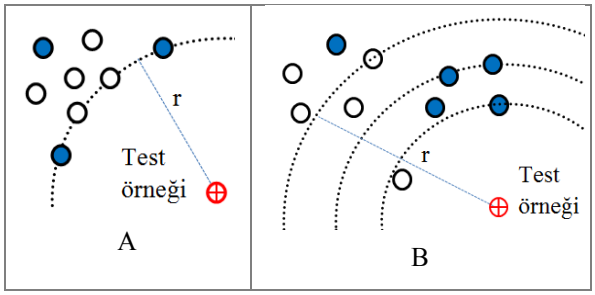
Şekil 2. Bazı veri setlerinin histogram bilgileri

k-NN sınıflandırıcısında her bir test örneği için en uygun *k* parametresini seçen bir sınıflandırıcı algoritması üzerinde çalışıldı. Çalışmamızın 2. bölümünde bu algoritmaya; 3. bölümünde bellek tabanlı sınıflandırıcılarda kullanılan arama yöntemlerine; 4. bölümünde elde edilen deneysel sonuçlara ve son bölümde ise değerlendirmelere yer verilmiştir.

2. En Yakın Küme ile Sınıflandırma

Sınıfı belirlenmek istenen bir test noktası etrafında hemen hemen aynı uzaklıkta birden fazla nokta bulunabilir. Bu durumda *k*-NN sınıflandırıcısı için seçilen *k* parametresinin 1 alınması durumunda rastgele seçimden ötürü sınıflandırma işlemi güvenilirliğini yitirmektedir. Şekil-3A'da görüldüğü üzere rastgele yapılan bu işlemde sınıf etiketi her denemede başka çıkabilmektedir.

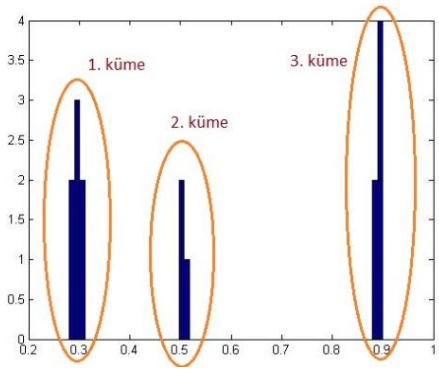
Bir test örneğine aynı uzaklıkta birden fazla örneğin bulunma ihtimali oldukça azdır. Fakat benzer uzaklıkta birden fazla örnek bulunabilmektedir. Şekil-3B'deki senaryoda *k* değerinin 1 veya daha fazla alınması durumunda sonuç farklı çıkacaktır.



Şekil 3. Sınıflandırma örnekleri

Bütün bunlar, *k*-NN algoritmasında her bir test örneği için en uygun *k* parametresinin seçilmesi gerektiğini göstermektedir. Çalışmamızda *k*-NN algoritmasına benzeyen fakat daha yüksek doruluk oranına sahip olabileceği düşünülen bir sınıflandırıcı üzerinde çalışıldı. En yakın küme sınıflandırıcısı (*One Nearest Cluster*, 1NC) diye isimlendirilen bu öğrenici, test noktasına en yakın ilk kümedeki örneklerin tamamını hesaplamaya katmaktadır. 1NC tekniğinin işlem basamakları şu şekildedir:

1. *N*, veri setinde örnek sayısı olmak üzere, veri setine ait tüm değerleri normalize et,
2. Test noktasına en yakın *M* adet örneği al ve *l* adet kümeye böl,
3. Test noktasına en yakında noktanın ait olduğu kümenin tüm elemanlarını *k*-NNyöntemiyle sınıflandırma işlemine al.



Şekil 4. Kümeleyerek sınıflandırma (1NC) örneği

Şekil-4'teki örnek senaryoda test noktana değişik uzaklıkta olan noktaların histogram grafiği görülmektedir. Bu noktalar kümeleme yöntemiyle (*clustering*) 3 adet kümeye bölünmüştür. 1NC sınıflandırıcısı için en yakında bulunan küme içerisindeki 7 adet örnek hesaplamaya katılacaktır (*k*-NN için *k*=7 alınacaktır).

Test noktasına en yakın M adet örneğin l adet kümeye bölünmesi ve sadece en yakındaki ilk kümenin işleme dâhil edilmesi dinamik bir yapıyı oluşturmaktadır. l adet kümenin (k-means'deki k ifadesinin k-NN'deki k ile karıştırılmaması için l ifadesi tercih edilmiştir) her birinde yaklaşık olarak M/l adet eleman olduğu düşünülebilir. Normalde$k$-NN sınıflandırıcısı için kullanıcı tarafından seçilen $k$parametresi ile bizim yöntemimizdekiM/lkombinasyonunundenk olması şu formül ile sağlanabilir:

$$k\text{NN}'deki\ k \cong \frac{M}{l} \quad (1)$$

Bu sayede M/likilisiyle her bir test örneğinin sınıflandırılması için uygun bir k değeri hesaplanmış olmaktadır.

Üzerinde çalışılan 1NC tekniğinde k-NN ve k-means yöntemlerinin toplam zaman karmaşıklıkları olduğu için normal k-NN sınıflandırıcısına göre bir miktar daha maliyetlidir [3].

3. En Yakın Örnekleri Arama Yöntemleri

k en yakın komşuluk algoritması (k-NN) bellek tabanlı bir sınıflandırıcıdır ve sınıflandırma işleminde her bir test örneği için eğitim setinde ayrı ayrı arama yapılmaktadır. Bu durum hesaplama zamanını artırmaktadır. Uygulamamızda hesaplama zamanını düşürmek için iki tip arama algoritması kullanılmıştır.Literatürde bir veri setinde boyutsayısı 10'dan büyük ise tam kapsamlı arama (*Exhaustive Search*); 10'dan küçük ise kD-Tree veri yapısı ile yapılan arama yöntemini tavsiye edilmektedir [4]. Kullanılan arama yöntemleri algoritmanın başarısını etkilememektedir.

3.1. Tam kapsamlı arama

Tam kapsamlı arama (*Exhaustive Search*) ile hiçbir veri yapısı ve algoritma kullanılmadan Öklid uzaklığına göre test noktasına en yakın k tane örnek sıralı bir şekilde aranır. Bu arama yönteminin zaman karmaşıklığı oldukça yüksektir. Dveri setinin boyut sayısı, N de eleman sayısı olmak üzere algoritmanın Big-O notasyonuna göre istenilen bir elemanı bulmanın zaman karmaşıklığı $O(k * D * N)$'dir [5].

3.2. kD-Tree ile arama

BSP (*Binary Space Partitioning*) yöntemlerinden biri olan kD-Tree (*k Dimensional Tree*), ikili arama ağacı olan BST (*Binary Search Tree*) veri yapısının çok boyutlu türüdür. Bu veri yapısında sadece aranan bir elemanı bulma maliyeti $O(D * \log N)$; her hangi bir test noktasına en yakın M adet noktanın bulunması $O(D * M * \log N)$'dir. Tam kapsamlı aramada bu maliyet $O(D * M * N)$ olduğu için kD-Tree daha avantajlıdır.

4. Pratik Uygulama ve Deneysel Sonuçlar

Verileri normalize edilmiş, kayıp değerleri yer değiştirilmiş, nominal değerleri ikili sayısal değerlere dönüştürülmüş 36 adet UCI veri seti, MATLAB ortamında 5x2 çapraz geçerleme ile test edildi. Veri setlerinde arama yöntemi olarak kD ağaç veri yapısı ve tam kapsamlı arama yöntemlerinden uygun olanı kullanıldı. Tablo-2'te yaptığımız çalışmanın sonuçları verilmektedir. k-NN sınıflandırıcısı için örnek olarak k parametresi 5 alınmıştır ve tüm veri setleri için doğruluk sonuçları hesaplanmıştır. İki farklı sınıflandırma mekanizması tarafından elde edilen sonuçların istatistiksel anlamlılığını tespit etmek için T-Test yöntemi [6] tercih edilmiştir. T-Test sonuçları üç farklı değer içermektedir: *win* (başarılı), *loss* (başarısız) ve *tie* (eşit).

Tablo-2'nin ilk sütununda 1NN sınıflandırıcısının; ikinci sütunda k=5 alınarak elde edilen k-NN sınıflandırıcısının; üçüncü sütunda *en yakındaki küme* (1NC) sınıflandırıcısında doğruluk oranları görülmektedir. 1NC'de her bir test noktasına en yakın (M değeri) 20 eleman alınarak *k-means* kümeleme yöntemiyle (iterasyon sayısı=100 ve k parametresi=4 alındı) 4 adet kümeye (l değeri) bölünmüştür. Bu durumda 4 adet kümenin her birinde yaklaşık olarak 5'er adet örnek bulunduğu düşünülebilir. Bu durum k-NN için k=5 anlamına gelir.

k-NN (k=5) ile 1NC ($M/l = 20/4$) sınıflandırıcıları ile elde edilen doğruluk oranlarının yüzdelik artış-azalış oranları ve bu iki sınıflandırıcının T-Test sonuçları*kNN-1NC karşılaştırma* isimli sütununda görülmektedir. Bu karşılaştırmada %15'e kadar sınıflandırma başarısında artış olduğunu gözlemlenmiştir.Her iki sınıflandırıcının karşılaştırması sonucunda 8 adet veri kümesinde başarı (*win*) sağlanmış, 20'sinde değişme olmamış (*tie*) ve 8 tanesinde başarısız (*loss*) olunmuştur. Ayrıca 1NC yöntemi, 1NN ile karşılaştırıldığında daha başarılı sonuçlar elde edilmiştir: 12 adet *win*, 20 adet *tie* ve sadece 4 adet *loss*. Görüldüğü üzere 1NC algoritması ile bazı veri setlerde daha başarılı sonuçlar elde edilmiştir.

1NC için M/l ikililerinden *k*-NN'deki *k*=5 parametresine denk gelen 10/2, 15/3, 30/6, 50/10 ve 100/20 kombinasyonları ayrı ayrı denenmiştir. Denemelerde 20/4 ile elde edilenbenzer doğruluk oranlarına ve hemen hemen aynı sayıda *win*, *tie* ve *loss* sonuçlarına ulaşılmıştır. Bu sonuçlar üç farklı çıkarım yapılmasını sağlamıştır. Birincisi benzer sonuçları veren düşük değerlikli M/l ikilisinintercih edilmesi hesaplama süresinin kısalmasını sağlar. İkincisi *k*-NN'deki *k*'ya eşit olması için genel olarak *M*, *k*'nın3 katı alınabilir ve l değeri de 3'e sabitlenebilir ($k \cong M/l$ olduğunu hatırlayınız). Üçüncü çıkarımda ise M/l ikilisi tek bir parametre gibi düşünülebilir.

Ayrıca tüm veri setleri için *k*-NN'de *k* değeri, sırasıyla 1'den 100'e kadar alınarak sınıflandırma başarıları elde edilmiştir. 14 adet veri setinde en yüksek başarı, *k*=1 alındığında hesaplanmıştır. 1NN'in en iyi örnek tabanlı sınıflandırıcı olduğu veri kümelerinde 1NC yönteminin daha yüksek başarı elde etmesi mümkün değildirdiye düşünülmüştü. Fakat 1NC yönteminde M/l ikilisi 20/4 alındığında bazı veri setlerinde (*ionosphere, d159, sonar*) 1NN'e göre daha da yüksek başarılar elde edilmiştir. Diğer veri setlerinde (*autos, glass, vowel, anneal, col10, labor, letter, mushroom, segment, soybean, zoo*) ise *tie* sonucu alınmıştır. Bu durum bazı veri setler için 1NC'nin, en başarılı sonucu veren küçük *k* parametreli *k*-NN'den bile daha başarılı olabileceğini göstermektedir.

5. Değerlendirme

k-NN'de her bir örnek için sabit bir*k* değerinin kullanılmasına karşın çalışmamızda her bir örnek için en uygun ve farklı *k* değerlerinin kullanılması daha yüksek doğruluk oranları elde etmemizi sağladı. Çalışmamızda *k*-NN'de olduğu gibi tek bir parametre yardımıyla daha başarılı sonuçlar elde edildi. Diğer bir taraftan üzerinde çalıştığımız yöntemde, örnek tabanlı bir sınıflandırıcıya ilave olarak kümeleme yöntemi kullandığı için bir miktar zaman karmaşıklığında ve hesaplama süresinde artış oldu.

6. Kaynaklar

[1] Myatt, G.J, Making Sence of Data: A Practical Guide to Exploratory Data Analysis and Data Mining, Wiley, (2007). s. 176-181.

[2] Bache, K. & Lichman, M. UCI Machine Learning Repository http://archive.ics.uci.edu/ml Irvine, University of California, (2013).

[3] Myatt, G.J, Making Sence of Data: A Practical Guide to Exploratory Data Analysis and Data Mining, Wiley, (2007). s. 120-129.

[4] MATLAB R2014a Tutorial, KD Tree Searcher class, www.mathworks.com/help/stats/

[5] M.A.Weiss, Data Structures&Algorithm Analysis in C++, Pearson (2013), s 83-85, 614-618, 629.

[6] Demsar J., Statistical Comparisons of Classifiers over Multiple Data Sets, Journal of Machine Learning Research 7, (2006) s 1-30.

				kNN-1NC karşılaştır-ma		1NN-1NC karşılaştır-ma	
Veri seti	1NN	kNN k=5	1NC M/k=20/4	%'lik artış	T-Test	%'lik artış	T-Test
abalone	0.2023	0.2301	0.2249	-2.26	loss	11.15	win
anneal	0.9769	0.9584	0.9715	1.36	tie	-0.55	tie
audiology	0.6757	0.6852	0.6556	-4.32	loss	-2.98	loss
autos	0.6505	0.5723	0.6010	5.02	win	-7.61	loss
balnc.-scale	0.7894	0.8576	0.8547	-0.34	tie	8.27	win
breast-cancr	0.6643	0.7098	0.7084	-0.20	tie	6.64	win
breast-w	0.9548	0.9671	0.9557	-1.18	tie	0.09	tie
col10	0.7249	0.7072	0.7168	1.36	tie	-1.12	tie
colic	0.6957	0.7777	0.7380	-5.10	loss	6.09	win
credit-a	0.7901	0.8304	0.8043	-3.14	loss	1.80	tie
credit-g	0.6824	0.7176	0.7116	-0.84	tie	4.28	win
d159	0.9451	0.9404	0.9490	0.92	tie	0.42	tie
diabetes	0.6943	0.7286	0.7143	-1.97	tie	2.88	tie
glass	0.6713	0.6410	0.6644	3.65	win	-1.03	tie
heart-statlog	0.7467	0.8052	0.7785	-3.31	loss	4.26	win
hepatitis	0.7948	0.8361	0.8039	-3.86	loss	1.14	tie
hypothyroid	0.9125	0.9329	0.9289	-0.44	tie	1.79	tie
ionosphere	0.8598	0.8387	0.8701	3.74	win	1.19	tie
iris	0.9467	0.9613	0.9520	-0.97	tie	0.56	tie
kr-vs-kp	0.8891	0.8923	0.9260	3.78	win	4.15	win
labor	0.8732	0.8421	0.8316	-1.25	tie	-4.76	loss
letter	0.9438	0.9343	0.9416	0.78	tie	-0.23	tie
lymph	0.7592	0.7915	0.7859	-0.71	tie	3.52	win
mushroom	1.0000	0.9999	0.9998	-0.01	tie	-0.02	tie
prim.-tumor	0.3874	0.4430	0.4291	-3.14	loss	10.77	win
ringnorm	0.7257	0.6623	0.7354	11.03	win	1.33	win
segment	0.9580	0.9443	0.9539	1.01	tie	-0.43	tie
sick	0.9569	0.9598	0.9562	-0.38	tie	-0.07	tie
sonar	0.8375	0.7481	0.8433	12.72	win	0.69	tie
soybean	0.8916	0.8776	0.8806	0.34	tie	-1.23	tie
splice	0.7357	0.7285	0.7628	4.71	win	3.68	win
vehicle	0.6723	0.6825	0.6752	-1.07	tie	0.43	tie
vote	0.9228	0.9297	0.9264	-0.35	tie	0.39	tie
vowel	0.9473	0.7806	0.8994	15.22	win	-5.05	loss
waveform	0.7278	0.7875	0.7476	-5.06	loss	2.72	win
zoo	0.9987	0.9929	0.9762	-1.68	tie	-2.25	tie

Tablo 2: Algoritmaların 36 veri kümesinde karşılaştırılması