İSTATİSTİK ÇALIŞMALARI

Editör: Doç.Dr. Fatma Sevinç KURNAZ

yaz
yayınları

İSTATİSTİK ÇALIŞMALARI

Editör

Doç.Dr. Fatma Sevinç KURNAZ

yaz yayınları

2024

yaz
yayınları

İSTATİSTİK ÇALIŞMALARI

Editor: Doç.Dr. Fatma Sevinç KURNAZ

E_ISBN 978-625-6642-95-9

Temmuz 2024 – Afyonkarahisar

Dizgi/Mizanpaj: YAZ Yayınları

Kapak Tasarım: YAZ Yayınları

YAZ Yayınları. Yayıncı Sertifika No: 73086

M.İhtisas OSB Mah. 4A Cad. No:3/3
İscehisar/AFYONKARAHİSAR

www.yazyayinlari.com

yazyayinlari@gmail.com

info@yazyayinlari.com

İÇİNDEKİLER

SEYREK KÜMELEME VE R UYGULAMALARI

Furkan YETER[1]

1. GİRİŞ

Günümüzde gelişen teknolojiler ve sensörler ile beraber her şeyi kaydederek veri haline getirebilmekteyiz. Bu doğrultuda elimizde devasa veri setleri biriktirebiliriz. Bu verilerin analiz edilmesi ve karmaşık yapı içerisinden anlamlı ve kullanılabilir bilgiler çıkarılması ihtiyaç haline gelmiştir.

Elimizdeki veri setinin karşılığı olarak oluşan sonuç bilgisi de elimizde varsa bu tarz veri setlerinde denetimli öğrenme dediğimiz içerisinden hem bağımsız x değişkenlerini hem de bağımlı y değişkenini bulunduran çalışma modelleri kullanılabilir. Fakat eğer elimizde bir sonuç bilgisi yoksa yani bağımlı değişken bilgisi yoksa ihtiyaçımız olan bilgiyi bağımsız x değişkenlerinden çıkarmalıyız. Bu tarz çalışmalarda ise denetimsiz öğrenme yöntemleri kullanılabilir.

Bu çalışmada denetimsiz öğrenme tekniklerinden biri olan kümeleme yöntemlerini ele alacağız. Kümeleme, veri madenciliği ve veri analizi alanlarında kullanılan güçlü bir tekniktir. Temel olarak, benzer özelliklere sahip veri noktalarını gruplayarak veri kümeleri oluşturmayı amaçlar. Bu gruplar, verinin içindeki yapıları ve ilişkileri anlamamıza yardımcı olur ve veri setlerinin daha iyi anlaşılmasını sağlar. Kümeleme yöntemi içerisinde bulunan K-Ortalamalar Kümeleme yöntemi ve Hiyerarşik Kümeleme yöntemini ve bu kümeleme yöntemlerinin

[1] Öğrenci(YL), Yıldız Teknik Üniversitesi, Fen Edebiyat Fakültesi, İstatistik Bölümü, furkann.yeter@gmail.com, ORCID: 0009-0005-4368-9121.

seyrek(sparse) modellerini açıklayıp, bir similasyon çalışması gerçekleştireceğiz.

2. K-ORTALAMALAR KÜMÜLEME YÖNTEMİ

K-Ortalamalar Kümeleme yöntemi veri madenciliği ve veri analizi alanlarında sıkça kullanılan bir yöntemdir. Temel amacı uzayda bulunan veri noktaları arasındaki benzerliği ve uzaklığı ölçümleyerek kümeler oluşturmaktır. Basit ve anlaşılabilir bir yöntemdir uygulaması kolaydır, bu sebeple en sık kullanılan kümeleme algoritmalarından biridir.

Temel olarak K-Ortalamalar Kümeleme, belirli bir K değeri seçilir ve rastgele K adet merkez nokta başlangıçta atanır. Daha sonra, n nesneyi, her nesnenin en yakın ortalamaya sahip kümeye ait olduğu k kümeye ayırmayı amaçlar. Her veri noktası, en yakın merkeze atanır ve bu işlem tekrarlanarak merkezlerin konumu yeniden hesaplanır. N adet nesnenin küme merkezlerine olan uzaklığı genellikle öklid uzaklığı formülü ile hesaplanır, fakat bunun yerine farklı uzaklık formülleri de kullanılabilir. Bu yöntem, mümkün olan en büyük farklılığa sahip tam olarak k farklı küme üretir. Bu adımlar, merkezlerin hareket etmeyeceği ve kümeleme sonuçlarının optimize edileceği bir noktaya kadar devam eder. K-Ortalamalar kümelemenin amacı, toplam küme içi varyansı veya karesel hata fonksiyonunu en aza indirmektir.

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i - c_j \|^2$$

Burada J bizim amaç fonksiyonumuzu temsil eder, k küme sayısını ve n elimizde bulunan satır sayısını temsil eder.

Peki bir k-ortalamalar algoritması nasıl çalışır şimdi aşağıda buna değinelim;

1) İlk olarak kaç adet merkez olacağı belirlenir ve buna k değeri denir.
2) Küme merkezi olarak rastgele k tane nokta seçilir.
3) Öklid mesafe fonksiyonuna göre nesnelerin k tane küme merkezine olan uzaklıkları hesaplanır ve en yakın kümeye atanır.

$$min \sum_{j=1}^{k} \sum_{x \in n_i} \| x - c_j \|^2$$

Burada k yine küme sayısını, c_j küme merkezlerini ve x gözlem değerlerini temsil eder.

4) Bütün nesneler bir kümeye atandıktan sonra, Her kümedeki tüm nesnelerin ortalamasını hesaplanır ve yeni küme merkezi olarak seçilir.

$$c_j = \frac{1}{|n_i|} \sum_{x \in n_i} x$$

5) Ardışık turlarda her bir kümeye aynı noktalar atanana kadar 2, 3 ve 4. adımları tekrarlanır.

Sonuç olarak, K-means kümeleme basitliği ve etkinliği ile ön plana çıkan güçlü bir kümeleme algoritmasıdır. Veri analizinde yaygın olarak kullanılan bu yöntem, veri içindeki yapıları keşfetmek ve anlamak için önemli bir araçtır. Ancak, uygulama alanına bağlı olarak, algoritmanın sınırlamaları ve zorlukları da göz önünde bulundurulmalıdır.

3. SEYREK K-ORTALAMALAR KÜMÜLEME YÖNTEMİ

Kümele çalışması yapacak olduğumuz veri boyut olarak n < p olabilir yani satır sayımız sütun sayımızdan küçük olabilir. Burada n gözlem sayısını p ise değişken sayısını temsil

etmektedir. Geleneksel K-Ortalamalar algoritması, her bir veri noktasını tam bir özellik vektörü olarak ele alır ve bu vektörler arasındaki benzerlikleri kullanarak kümeleme yapar. Ancak, bu yaklaşım büyük boyutlu ve düşük yoğunluklu özellik uzaylarında, yani "seyrek" veri kümelerinde etkili olmayabilir.

Seyrek veri kümeleri, birçok boyutta büyük özellik vektörlerine sahip olabilir, ancak her bir özellik vektöründe sadece birkaç özellik (veya boyut) önemlidir; diğerleri genellikle sıfırdır. Bu tür veri kümelerinde, geleneksel K-Ortalamalar algoritması etkili bir şekilde çalışamaz çünkü sıfır olmayan özelliklerin yoğun olmadığı boyutlarda kümeleme yapmak zordur.

Seyrek K-Ortalamalar, bu tür seyrek veri kümeleri için özel olarak tasarlanmıştır. Bu algoritma, sıfır olmayan özelliklere odaklanır ve sıfır olmayan özelliklerin yoğunluğunu dikkate alarak kümeleme yapar. Bu şekilde, seyrek veri kümelerinde daha iyi performans gösterebilir ve daha anlamlı kümelemeler elde edebilir.

Geleneksel K-Ortalamalar algoritmasında amaç küme içindeki toplam varyasnı minimize etmektir, bu doğrultuda kümeler arası varyansı maksimize etmekte aynı hedeftir.

K-ortalamalar kümelemesi, küme içi kareler toplamını (WCSS) en aza indirir. Yani, n gözlemi K kümeye veya kümelere ayırmaya çalışır, öyle ki WCSS minimumdur.

$$\sum_{k=1}^{K} \frac{1}{n_k} \sum_{\mathrm{i,i'} \in C_k} \sum_{j} \mathrm{d_{i,\,i',\,j}}$$

Burada n_k, K kümesindeki gözlem sayısını, C_k ise K kümesindeki gözlemlerin indislerini temsil etmektedir. di,i',j ise burada j kümesine ait i ve i′ gözlemleri arasındaki benzerlik ölçüsünü ifade etmektedir. Küme içindeki kareler toplamını bu çalışma boyunca $di,i',j = (\mathrm{Xij} - \mathrm{Xi'j})^2$

olarak hesaplayacağız. Buna ek olarak kümeler arası varyasnı ise aşağıdaki şekilde hesaplayabiliriz.

$$\sum_{j=1}^{p}\left(\frac{1}{n}\sum_{i=1}^{n}\sum_{i'=1}^{n} d_{i,i',j} - \sum_{k=1}^{K}\frac{1}{n_k}\sum_{i,i'\in C_k} d_{i,i',j}\right)$$

Eğer küme içindeki varyasnsı (wcss) minimize edersek , aynı şekilde kümeler arası varyansı (bcss) maksimize etmiş oluruz.

Kümeler arası varyansı optimize ederken bir ağırlıklandırma işlemi ekleyerek seyrek değişkenler üzerinde kısıtlamalar yapabiliriz bu sayede Seyrek K-Ortalama modeli oluşturabiliriz.

$$maximize\ C_1,\dots,C_K,W\ \left\{\sum_{j=1}^{p} w_j\left(-\sum_{k=1}^{K}\frac{1}{n_k}\sum_{i,i'\in C_k} d_{i,i',j}\right)\right\}$$

olduğu durumda $||w||^2 \le 1, ||w||_1 \le s,\ w_j \ge 0\ \forall\ j$

Burada w_j, j özelliğine(değişkenine) karşılık gelen bir ağırlıktır. s ise bir ayarlama parametresidir. Ağırlıklı toplamın her bir elemanı negatif olduğundan, maksimum değer, s değerinden bağımsız olarak tüm ağırlıklar sıfır olduğunda ortaya çıkar.

w üzerindeki L1 veya Lasso cezası, ayarlama parametresi s 'nin küçük değerleri için seyreklikle sonuçlanır: yani, w_j'lerin bazıları sıfıra eşit olacaktır. L2 cezası da önemli bir rol oynar, çünkü bu ceza olmadan w 'nin en fazla bir elemanı genel olarak sıfır olmayacaktır.

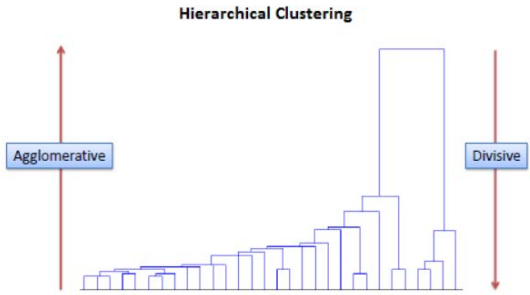
w_j değeri, j özelliğinin ortaya çıkan seyrek kümelemeye katkısı olarak yorumlanabilir: w_j'nin büyük bir değeri, büyük katkıda bulunan bir özelliği gösterir ve w_j = 0, j özelliğinin kümelemeye dahil olmadığı anlamına gelir.

4. HİYERARŞİK KÜMELEME

Hiyerarşik kümeleme K-Ortalamalar kümesi kadar popüler ve sık kullanılan bir kümeleme yöntemidir. Bu yöntemde K-Ortalamalar yönteminin tersine başlangıçta bir küme sayısı(k) belirlenmez. Buradaki temel amaç benzer özniteliklerin bir araya gelmesine veya tam tersine bölünmesine dayanmaktadır.

Bu işlemi yapabilmek için iki farklı çalışma şekli vardır. Birincisi birleştirici (agglomerative) ikincisi ise bölücü (divisive) yöntemleridir. Tüme varım (bottom up) olarakta bilinen birleştirici yöntemindeki temel mantık başlangıçta n adet nesneler birbirinden ayrı olarak uzayda yer almaktadır, yani her bir nesne ayrı bir küme görevi görür, ardından birbiri ile benzer olan kümeler birleştirilir ve en son tek bir küme oluşturulana kadar devam eder. Tümden gelim (top bottom) olarak bilinen bölücü yöntemindeki temel mantık ise n adet gözlemin başlangıçta tek bir kümede yer almasıdır, ardından adım adım uzaklık/benzerlik değerlerine göre nesneler ana kümeden ayrılır, ve farklı kümeler oluşturulur, en son aşamada n adet gözlem n adet kümeyi temsil eder.

Şekil 1. Hiyerarşik Kümelemenin İki Farklı Yöntemi



Hiyerarşik kümeleme, yeni kümeler oluşturmak için bir uzaklık/benzerlik ölçüsü kullanır. Birleştirici yöntem ile kurulan bir kümeleme modeli adım adım şu şekilde çalışır:

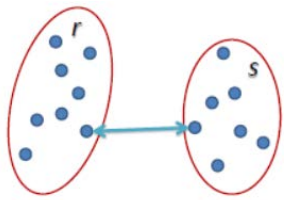
1) Belirli bir mesafe metriği kullanarak farklılaşma matrisi hesaplanır
2) Her veri noktası bir kümeye atanır
3) Kümeler arasındaki benzerlik için bir metriğe dayalı olarak kümeler birleştirilir
4) Uzaklık matrisi güncellenir
5) Yalnızca tek bir küme kalana kadar Adım 3 ve Adım 4 tekrarlanır

Noktalar arası uzaklığı genllikle öklid uzaklığı ile ölçeriz fakat bunun yerine farklı uzalık ölçütleri de kullanılarbilir. Kümeler arsındaki mesafeleri ölçerken bir kaç farklı yöntem kullanılabilir. Aşağıda kısaca bu yöntemlere bakabiliriz.

1) Tek Bağlantı

Tek bağlantılı hiyerarşik kümelemede, iki küme arasındaki kısa mesafe olarak tanımlanır. Örneğin, "r" ve "s" kümeleri arasındaki mesafe, en yakın iki noktaları arasındaki okun uzunluğuna eşittir.

Şekil 2. Tek Bağlantı Yönteminin Görsel Gösterimi

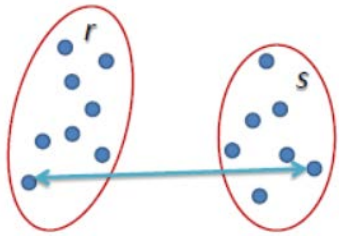


$$L(r,s) = \min\left(D\left(x_{ri}, x_{sj}\right)\right)$$

2) Komple Bağlantı

Tam bağlantılı hiyerarşik kümelemede, iki küme arasındaki en uzun mesafe olarak tanımlanır. Örneğin, "r" ve "s" kümeleri arasındaki mesafe, en uzak iki noktaları arasındaki okun uzunluğuna eşittir.

Şekil 3. Komple Bağlantının Görsel Gösterimi

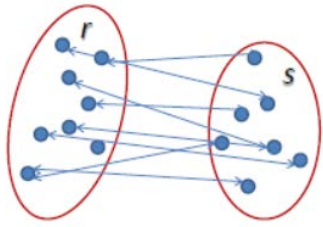


$$L(r,s) = \max\left(D\left(x_{ri}, x_{sj}\right)\right)$$

3) Ortalama Bağlantı

Ortalama bağlantı hiyerarşik kümelemede, iki küme arasındaki mesafe bir kümedeki her noktanın diğer kümedeki her noktaya olan ortalama mesafesi olarak tanımlanır. Örneğin, "r" ve "s" kümeleri arasındaki mesafe, bir kümenin noktalarını diğerine bağlayan her bir okun ortalama uzunluğuna eşittir.

Şekil 4. Ortalama Bağlantının Görsel Gösterimi



$$L(r,s) = \frac{1}{n_r n_s}\sum_{i=1}^{n_r}\sum_{j=1}^{n_s} D\left(x_{ri}, x_{sj}\right)$$

Bu yöntemlerden birini kullanarak kümele işlemini tamamlayan hiyerarşik kümeleme en son bir dendogram çıkararak grafiksel bir gösterim de sağlamaktadır.

Ayrıca, hiyerarşik kümeleme, diğer kümeleme yöntemleriyle birlikte kullanılarak veri içindeki yapıları daha iyi anlamak için bir araç olarak da kullanılabilir.

Hem basitliği hem de esnekliği nedeniyle, hiyerarşik kümeleme, veri analizinde yaygın olarak kullanılan bir tekniktir. Her ne kadar büyük veri kümeleri üzerinde çalışırken hesaplama açısından maliyetli olabilirse de, genelde veri içindeki yapıları anlamak için kapsamlı bir yaklaşım sunar.

5. SEYREK HİYERARŞİK KÜMELEME

Hiyerarşik kümele yaparken her zaman bütün değişkenleri kullanmak istemeyebiliriz ya da bazı değişkenler kullanılmaya uygun olmayabilir bu noktada seyrek hiyerarşik kümeleme yöntemi ile daha sağlıklı ve güvenilir bir hiyerarşik kümelemesi yapabiliriz.

Standart Hiyerarşik Kümeleme kısmında anlaltığımız gibi bu algoritma kümeleme işlemleri sırasında ilk olarak bir farklılaşma matrisi kullanır ve bu farklılaşma değerlerine göre kümeleri birleştirerek veya bölerek kümeleme işlemini tamamlar.

Standart hali için farklılaşma matrisi aşağıdaki şekilde öklid uzaklığı yöntemi ile oluşturulur.

$$D_{i,i'} = \sum_{j=1}^{p} d_{i,\,i',\,j}$$

$$d_{i,i',j} = (X_{ij} - X_{i'j})^2$$

Burada $D_{i,i'}$ farklılaşma matrisini temsil eder. $d_{i,i',j}$ ise bu matrisin hesaplandığı uzaklık ölçütüdür.

Eğer biz bu farklılaşma matrisini bir wj ağırlığı ile ağırlıklandırırsak ve kümeleme hesaplamalarını buna göre hesaplarsak seyrek hiyerarşik kümeleme yöntemini uygulamış oluruz.

$$\widetilde{D}_{i,i'} = \sum_{j=1}^{p} w_j d_{i,i',j}$$

Burada eklemiş olduğumuz w_jağırlık katsayısının ilgili olan j sütunu önemini yansıtmasını sağlıyoruz. Bu farklılaşma matrisini bulmak için aşağıdaki matris ayrışımını uygulamamız gerekmektedir.

$$maximize\ \{u^T \Delta w\}$$

olduğunda $u \in R^{N^2}, w \in R^p$ ve $||u||_2 \leq 1, \left|\left|w\right|\right|_2 \leq 1, \left|\left|w\right|\right|_1 \leq s, w \geqslant 0$

Burada $\Delta\ N^2 xp$ 'lik bir matrisi temsil eder, bu matris j sütunu için çift yönlü farklılaşma matrisidir.

Aynı şekilde w ise ağırlık vektörüdür $w_j,\ j \in \{1, \ldots, p\}$

Bu matrisin tekil değer ayrışımı yöntemi ile çözülmesi ile elde edilen matris bize ağırlıklandırılmış farklılaşma matrisini verir. Bu ağırlıklandırılmış farklılaşma matrisi ile beraber seyrek hiyerarşik kümeleme yöntemini uygulamış oluruz.

6. R UYGULAMALARI

Seyrek kümeleme uygulamaları için R programlama dili oldukça kullanışlıdır. Algoritma hakkında doküman ihtiyaçını da diğer programlama dillerine göre daha iyi karşılamaktadır

R da seyrek kümele için öncelikle "sparcl" paketinin yüklenmesi gerekmektedir. Bunun için R konsolunda aşağıdaki kod çalıştırılmalıdır.

```
install.packages("sparcl")
```

Ardından bu paketi kullanarak seyrek kümeleme algoritmalarını kullanabiliriz.

6.1.Seyrek K-Ortalamalar Kümeleme ve Klasik K-Ortalamalar Kümeleme

Bu çalışmada Hartigan'ın (1975) Dünyanın Başlıca Şehirleri Arasındaki Havayolu Mesafesi adlı veri setini kullanacağız. Tablo, dünyanın belli başlı şehirleri arasındaki havayolu mesafelerini yüzlerce mil cinsinden içermektedir. Veri seti içerisinde 30 farklı ülkenin birbirlerine olan uzaklık değerlerini içermektedir.

İlk olarak üzerinde çalışacağımız veri matrisini normalize ediyoruz bunun için scale komutunu kullandık.

```
scaleair <- scale(x, TRUE, TRUE)
```

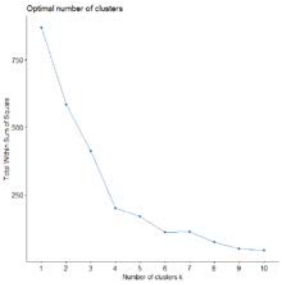
x: İşlem yapmak istediğiniz veri. TRUE (center): Bu parametre TRUE olduğu için, x'in her bir sütununun ortalaması çıkarılacaktır. TRUE (scale): Bu parametre TRUE olduğu için, x'in her bir sütununun standart sapmasına bölünecektir.

Bu parametreler sayesinde ortalması 0 standart sapması 1 olan bir normal dağılım elde edilmiş olur.

```
fviz_nbclust(scaleair, kmeans, method = "wss")
```

Kodu ile beraber klasik kümele yöntemi ve dirsek yöntemini uygulayarak uygun küme sayısını tespit edebiliriz. Şekil 5'e bakarsak dirseğin tam 4 noktasından kırıldığını görebiliriz. Bu sebeple bundan sonraki adımlarda küme sayısı 4 olarak kullanılacaktır.

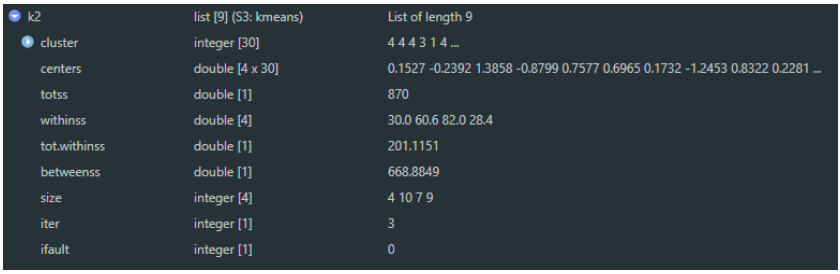
Şekil 5. K-Ortalamalar Yöntemi ile Oluşturulmuş Dirsek Grafiği



```
k <- kmeans(scaleair, centers = 4, nstart = 25)
```

Kodu ile beraber klasik k-ortalamalar kümüleme yapılabilir ve çıktı olarak şekil 6'daki gibi atanan kümeler, küme merkezleri ve performans metrikleri görüntülenebilir.

Şekil 6. K-Ortalamalar Kümelemesini Sonuçları



Bu işlemlerin ardından "KMeansSparseCluster.permute" fonksiyonu ile seyrek (sparse) veri kümeleri için K-means kümeleme yapar ve ayrıca tuning parametresi olan wbounds için en uygun değeri belirlemek için permütasyon testlerini yaparız.

```
km.perm <- KMeansSparseCluster.permute(scaleair, K=4,
wbounds=seq(3,7,len=15), nperms=25)
```

x: Kümelenecek veri seti, gözlem ve özelliklerden oluşan bir matristir.

K: K-means algoritmasının kaç küme oluşturacağını belirtir.

wbounds: K-means algoritmasının ceza (penalty) parametresidir ve belirli bir aralıkta (3 ile 7 arasında) 15 farklı değer alacak şekilde tanımlanmıştır. Bu, her bir özellik için farklı ağırlıklar belirlemek ve modelin karmaşıklığını kontrol etmek içindir.

nperms: Bu parametre, permütasyon testlerinin kaç kez yapılacağını belirtir. Bu durumda, 25 kez permütasyon yapılacaktır. Permütasyon yaklaşımı (permutation approach), bir modelin veya değişkenlerin önemini değerlendirmek ve tuning parametrelerini seçmek için kullanılan güçlü bir istatistiksel

yöntemdir. Bu yöntem, verilerin rastgele karıştırılmasını (permütasyonunu) ve bu permütasyonlarla modelin performansının değerlendirilmesini içerir.

Permütasyon yaklaşımının adımları şu şekildedir;

1) Orijinal Modeli Eğitme ve Performansı Ölçme: İlk olarak, orijinal veri seti kullanılarak model eğitilir. Bu modelin performansı bir değerlendirme metriği (örneğin, doğruluk, hata oranı, R^2) ile ölçülür.
2) Verileri Permüte Etme:Veri setindeki bağımsız değişkenler (özellikler) veya bağımlı değişken (hedef) rastgele karıştırılır. Bu, her bir özelliğin gerçek verilerden ne kadar sapabileceğini anlamak için yapılır.
3) Permütasyonlu Verilerle Modeli Eğitme: Karıştırılmış veri setleri ile model tekrar tekrar eğitilir. Her permütasyon için model performansı ölçülür ve kaydedilir.
4) Performansın Karşılaştırılması: Orijinal model performansı ile permütasyonlu modellerin performansları karşılaştırılır. Eğer permütasyonlu verilerle eğitilen modellerin performansı, orijinal modelin performansına yakınsa, bu, özelliklerin model performansında önemli bir rol oynamadığını gösterebilir.
5) Özelliklerin Önemi ve Tuning Parametrelerinin Seçimi: Özelliklerin önem derecesi, permütasyonlu verilerle model performansında oluşan değişikliklere göre belirlenir. Tuning parametreleri, permütasyon sonuçlarına göre optimize edilir; en iyi performansı veren parametreler seçilir.

KMeansSparseCluster.permute fonksiyonu ile kurulan modellerin çıktısı şu şekil 7'deki gibi yazdırılır.

```
print(km.perm)
```

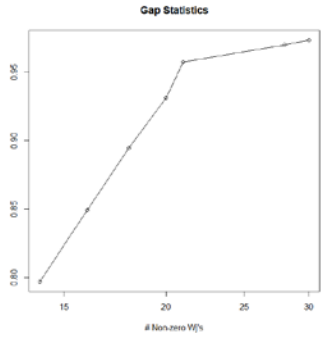
Şekil 7. Kurulan Seyrek K- Ortalamalar Modellerinin Performans Metrikleri ve Sonuçları

```
Tuning parameter selection results for Sparse K-means Clustering:
   Wbound # Non-Zero W's Gap Statistic Standard Deviation
1  3.0000            14         0.7969             0.0802
2  3.2857            16         0.8490             0.0782
3  3.5714            18         0.8944             0.0779
4  3.8571            20         0.9313             0.0779
5  4.1429            21         0.9576             0.0779
6  4.4286            28         0.9700             0.0779
7  4.7143            30         0.9735             0.0779
8  5.0000            30         0.9735             0.0779
9  5.2857            30         0.9735             0.0779
10 5.5714            30         0.9735             0.0779
11 5.8571            30         0.9735             0.0779
12 6.1429            30         0.9735             0.0779
13 6.4286            30         0.9735             0.0779
14 6.7143            30         0.9735             0.0779
15 7.0000            30         0.9735             0.0779
Tuning parameter that leads to largest Gap statistic:  4.714286
```

Yine KMeansSparseCluster.permute fonksiyonu ile kurulan modellerin çıktısı şu şekilde görüntülenebilir.

```
plot(km.perm)
```

Şekil 8. Ağırlıklara Göre Elde Edilen Gap İstatistikleri



Şekil 8'de verilen ağırlıklara göre değişkenlerin almış olduğu kat sayı ve buna karşılık gelen GAP istatistiğini görüyoruz. Gap istatistiği, kümeleme analizinde kullanılan bir

yöntemdir. Bu yöntem, belirli bir tuning parametresinin (ayarlama parametresi) performansını değerlendirmek için kullanılır. Temel olarak, orijinal verilerle ve permütasyonla (karıştırılmış) verilerle elde edilen sonuçlar arasındaki farkı ölçer.

$$Gap(s) = \log\big(O(s)\big) - mean(\log(O * (s)))$$

$O(s)$: Orijinal veri seti için tuning parametresi s kullanılarak hesaplanan amaç fonksiyonu değeri.

$\log\big(O(s)\big)$: Bu değerin logaritması.

$mean(\log(O * (s)))$: Permütasyonlu veri seti için hesaplanan logaritma değerlerinin ortalaması

Bu istatistiğin hesaplanması ise şu şekildedir;

1) Tuning parametresi s için, orijinal veri seti kullanılarak $O(s)$ hesaplanır.
2) Veri seti birçok kez (örneğin, nperms = 5) permüte edilir. Her permütasyon için amaç fonksiyonu $O * (s)$ hesaplanır. Bu değerlerin logaritması alınır ve ortalaması hesaplanır
3) Orijinal veri seti için amaç fonksiyonunun logaritması alınır. Bu değerden, permütasyonlu veri seti logaritmalarının ortalaması çıkarılır.

Sonuç olarak Gap istatistiği, doğru tuning parametresini seçmek için kullanılır. Eğer gap değeri büyükse, bu tuning parametresi verilerle iyi bir uyum sağlıyor demektir. Eğer gap değeri küçükse veya negatifse, bu tuning parametresi iyi bir sonuç vermiyor demektir.

Daha sonra aşağıdaki kod ile beraber kurulan modellen arasında en başarılı olanı çağırabiliriz ve sonuçlarını yazdırabiliriz.

```
km.out <-
KMeansSparseCluster(scaleair,K=4,wbounds=km.perm$bestw)

print(km.out)
```

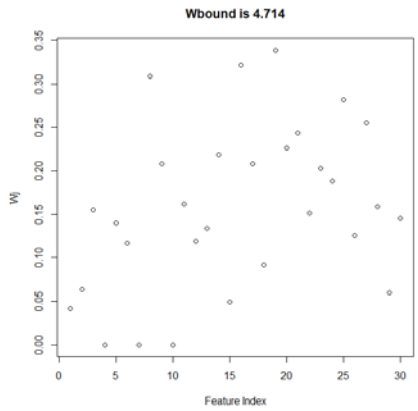
Şekil 9. En Başarılı Gelen Ağırlık Katsayısı Ve Sonuçları

```
Wbound is  4.714286 :
Number of non-zero weights:  27
Sum of weights:  4.714382
Clustering:  1 1 1 2 3 1 3 4 2 4 1 4 1 2 2 4 4 1 4 4 4 1 3 1 4 3 4 2 2 2
```

Bu işlem sonucunda en başarılı gelen ağırlığın 3.85 olduğu görülmüştür. Bu ağırlık sonucunda oluşan değişken ağırlık grafiği ise şekil 10'daki gibi görüntülenebilir.

```
plot(km.out)
```

Şekil 10. Değişkenlerin Ağırlık Katsayıları



Şekil 11'de ise kurulan klasik kümeleme modelinin ve seyrek kümeleme modelinin sonuçlarını karşılaştırmaktayız. Çıktılara baktığımızda hem klasiik modelin hem de seyrek modelin aynı sonuçları elde ettiğini görmekteyiz, iki modelin performansı burada aynı çıktı fakat değişken kullanımı olarak bakarsak klasik yöntem 30 değişkeni de kullanırken, seyrek model 27 değişkeni kullanarak bu sonucu elde etti bu noktada değişken maliyetini göz önüne alırsak seyrek modelin daha iyi olduğunu söyleyebiliriz.

Şekil 11. Klasik ve Seyrek K-Ortalamalar Kümelemesinin Karşılaştırılması

```
Confusion Matrix and Statistics

          Sparse K-Means
K-Means  1  2  3  4
       1 9  0  0  0
       2 0  7  0  0
       3 0  0  4  0
       4 0  0  0 10

Overall Statistics

               Accuracy : 1
                 95% CI : (0.8843, 1)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : 4.857e-15

                  Kappa : 1

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4
Sensitivity             1.0   1.0000   1.0000   1.0000
Specificity             1.0   1.0000   1.0000   1.0000
Pos Pred Value          1.0   1.0000   1.0000   1.0000
Neg Pred Value          1.0   1.0000   1.0000   1.0000
Prevalence              0.3   0.2333   0.1333   0.3333
Detection Rate          0.3   0.2333   0.1333   0.3333
Detection Prevalence    0.3   0.2333   0.1333   0.3333
Balanced Accuracy       1.0   1.0000   1.0000   1.0000
```

Şimdi aynı işlemleri bir de farklı bir veri seti ile deneyelim. Bu sefer Colin Greensill'den (Mühendislik ve Fiziksel Sistemler Fakültesi, Central Queensland Üniversitesi, Rockhampton, Avustralya) elde edilen, aynı meyvenin altı farklı çeşidinin (kavun - Cucumis melo L. Cantaloupensis grubu) spektrumlarını içeren bir veri setini kullanacağız. Toplam veri seti 256 dalga boyunda ölçülen 2818 spektrum içeriyor. Açıklama amacıyla, sırasıyla 490, 106 ve 500 boyutlarında D, M ve HA olarak adlandırılan yalnızca üç çeşit dikkate alınmıştır. Veri seti

1096 gözlem ve 257 sütun içermektedir. Bir sütun kavun tiplerini temsil etmektedir ve D, M ve HA gibi değerler almaktadır.

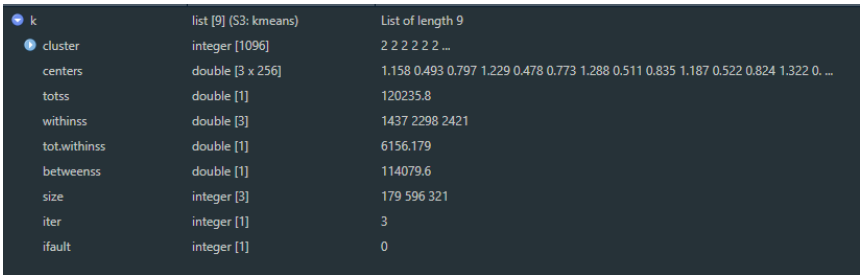
ilk olarak veriden tip sütununu çıkartıyoruz. Bunu aşağıdaki kod ile yapabiliriz.

```
fruit2 <- fruit[,-1]
```

Ardından 3 küme olacak şekilde klasik k-ortalamalar kümelemesi yapıp çıktısını görebiliriz.

```
k <- kmeans(fruit2, centers = 3, nstart = 25)
```

Şekil 12. K-ortalamalar kümelemesini sonuçları



k	list [9] (S3: kmeans)	List of length 9
cluster	integer [1096]	2 2 2 2 2 2 ...
centers	double [3 x 256]	1.158 0.493 0.797 1.229 0.478 0.773 1.288 0.511 0.835 1.187 0.522 0.824 1.322 0. ...
totss	double [1]	120235.8
withinss	double [3]	1437 2298 2421
tot.withinss	double [1]	6156.179
betweenss	double [1]	114079.6
size	integer [3]	179 596 321
iter	integer [1]	3
ifault	integer [1]	0

```
km.perm <- KMeansSparseCluster.permute(fruit2, K=3,
wbounds=seq(3,7,len=15), nperms=25)
```

KMeansSparseCluster.permute fonksiyonu ile kurulan modellerin çıktısı ve grafiği şu şekilde yazdırılır.

```
print(km.perm)
```

Şekil 13. Kurulan Seyrek K- Ortalamalar Modellerinin Performans Metrikleri ve Sonuçları

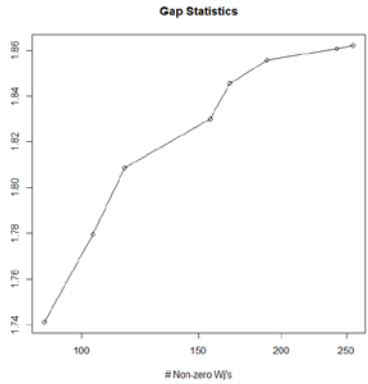
Tuning parameter selection results for Sparse K-means Clustering:

	Wbound	# Non-Zero W's	Gap Statistic	Standard Deviation
1	8.0000	88	1.7414	0.0514
2	8.6316	104	1.7793	0.0514
3	9.2632	116	1.8087	0.0514
4	9.8947	156	1.8300	0.0514
5	10.5263	167	1.8456	0.0514
6	11.1579	190	1.8558	0.0514
7	11.7895	242	1.8608	0.0514
8	12.4211	256	1.8620	0.0514

```
9  13.0526          256       1.8620          0.0514
10 13.6842          256       1.8620          0.0514
11 14.3158          256       1.8620          0.0514
12 14.9474          256       1.8620          0.0514
13 15.5789          256       1.8620          0.0514
14 16.2105          256       1.8620          0.0514
15 16.8421          256       1.8620          0.0514
16 17.4737          256       1.8620          0.0514
17 18.1053          256       1.8620          0.0514
18 18.7368          256       1.8620          0.0514
19 19.3684          256       1.8620          0.0514
20 20.0000          256       1.8620          0.0514
Tuning parameter that leads to largest Gap statistic:  12.42105
```

```
plot(km.perm)
```

Şekil 14. Ağırlıklara Göre Elde Edilen Gap İstatistikleri



Aşağıdaki kodlar ile beraber en iyi gelen seyrek modelin sonuçlarını görebiliriz.

```
km.out <-
KMeansSparseCluster(fruit2,K=3,wbounds=km.perm$bestw)

print(km.out)
```

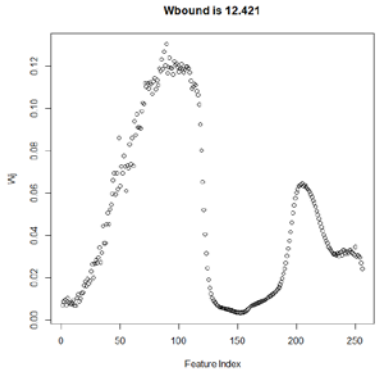
Şekil 15. En Başarılı Gelen Ağırlık Katsayısı ve Sonuçları

```
Wbound is  12.42105 :
Number of non-zero weights:  256
Sum of weights:  12.30917
Clustering:  1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 2 2 1 1 1
1 1 1 1 1 1 1 .... 1 1
```

Bu kod ile ise en iyi gelen modeldeli değişken katsılarını şekil 16'daki gibi görebilmekteyiz.

```
plot(km.out)
```

Şekil 16. Değişkenlerin Ağırlık Katsayıları



6.2.Seyrek Hiyerarşik Kümeleme

Bu kısımda da seyerk k-ortalamalar çalışmasında kullandığımız normalleştirilmiş veri setinu kullanacağız. İhtiyacımız olan fonksiyonlar "sparcl" paketinde bulunmaktadır. Bu sebeple tekrar yükleme yapmadan direkt kodları çalıştırabiliriz.

```
perm.out <- HierarchicalSparseCluster.permute(scaleair, wbounds=
c(1.5,2:6), nperms=5)
```

Bu kod ile önceki çalışmada olduğu gibi belli bir aralıkta yer alan farklı ağırlık değerleri ile beraber permütasyon yaklaşımı uygulayarak farlı modeller kurar. Burada farklı olarak ağırlık sınırları için 1.5 ve 2 ile 6 arasındaki tam sayılar kullanıldı (yani,

1.5, 2, 3, 4, 5, 6). Bu şekilde bir vektör vererek de ağırlıkları belirlemek mümkündür.

Modelin çıktısına ve gap grafiğine bakmak istersen aşağıdaki gibi bakabiliriz.

```
print(perm.out)
```

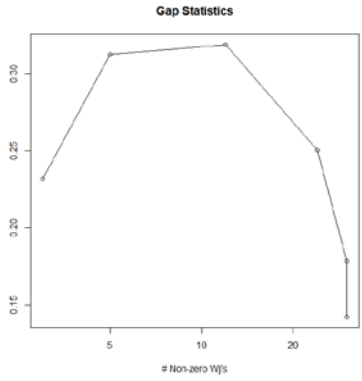
```
plot(perm.out)
```

Şekil 17. Kurulan Seyrek Hiyerarşik Modellerinin Performans Metrikleri ve Sonuçları

```
Tuning parameter selection results for Sparse Hierarchical Clustering:
  Wbound # Non-Zero W's Gap Statistic Standard Deviation
1    1.5             3       0.2320             0.0181
2    2.0             5       0.3125             0.0101
3    3.0            12       0.3187             0.0048
4    4.0            24       0.2503             0.0031
5    5.0            30       0.1787             0.0023
6    6.0            30       0.1425             0.0019
Tuning parameter that leads to largest Gap statistic:  3
```

Şekil 17'deki sonuçlara bakacak olursak en başarılı ağırlıklandırmanın 3 değerinde olduğunu görmekteyiz, yine aynı şekilde en büyük gap istatistiği değeri 3 değerinde yakalanmaktadır.

Şekil 18. Ağırlıklara Göre Elde Edilen Gap İstatistikleri



Bu işlemler ardından en başarılı olan modeli çağırabiliriz.

```
sparsehc <- HierarchicalSparseCluster(dists=perm.out$dists,
wbound=perm.out$bestw, method="complete")
```

Burada;

dists=perm.out$dists: Daha önce HierarchicalSparseCluster.permute işlevi ile hesaplanan mesafeleri kullanır. Bu mesafeler, permütasyon testi sırasında elde edilen mesafelerdir ve kümeleme algoritmasının girdi verisi olarak kullanılır.

wbound=perm.out$bestw: Permütasyon testi sonucunda belirlenen en iyi ağırlık sınırını kullanır. Bu, perm.out nesnesinin içinde saklanan ve permütasyon testinden elde edilen optimal ağırlık sınırıdır.

method="complete": Hiyerarşik kümelemenin bağlama metodunu belirtir. "complete" metodu, complete linkage clustering (tam bağlantı kümeleme) olarak bilinir ve iki küme arasındaki en uzak mesafeyi kullanarak kümeler arasındaki benzerlikleri belirler.

Ve son olarak oluşturulan optimal modelin çıktısını yazdırabilir ve grafiklerini aşağıdaki kod ile şekil 19'daki görebiliriz.

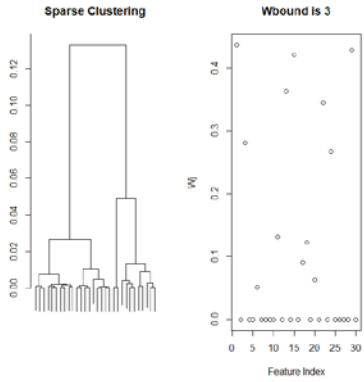
```
print(sparsehc)
```

```
plot(sparsehc)
```

Şekil 19. En Başarılı Gelen Ağırlık Katsayısı ve Sonuçları

```
Wbound is  3 :
Number of non-zero weights:  12
Sum of weights:  2.999987
```

Şekil 20. Denfogram Grafiği ve Değişkenlerin Ağırlık Katsayıları



Şekil 20 ile beraber oluşan dendogramı ve değişkenlerin almış olduğu ağırlık katsayılarını görüntülüyebiliriz.

Aynı çalışmaları bir de Colin Greensill'den elde edilen, aynı meyvenin altı farklı çeşidinin spektrumlarını içeren bir veri setini kullanarak uygulayacağız.

```
perm.out <- HierarchicalSparseCluster.permute(scaledfruit, wbounds=
c(1.5,2:6), nperms=5)
```

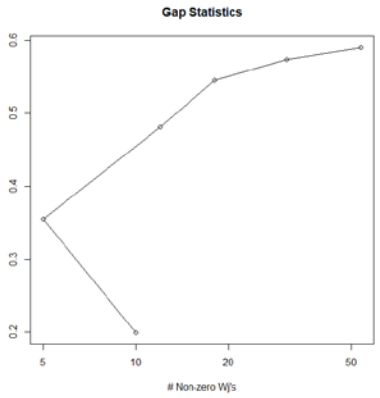
Modelin çıktısına ve gap grafiğine bakmak istersen aşağıdaki gibi bakabiliriz.

```
print(perm.out)

plot(perm.out)
```

Şekil 21. Kurulan Seyrek Hiyerarşik Modellerinin Performans Metrikleri ve Sonuçları

```
Tuning parameter selection results for Sparse Hierarchical Clustering:
  Wbound # Non-Zero W's Gap Statistic Standard Deviation
1    1.5          10       0.1996             5e-04
2    2.0           5       0.3548             4e-04
3    3.0          12       0.4816             1e-03
4    4.0          18       0.5452             9e-04
5    5.0          31       0.5738             7e-04
6    6.0          54       0.5900             9e-04

          Tuning parameter that leads to largest Gap statistic:  6
```

Şekil 22. Ağırlıklara Göre Elde Edilen Gap İstatistikleri



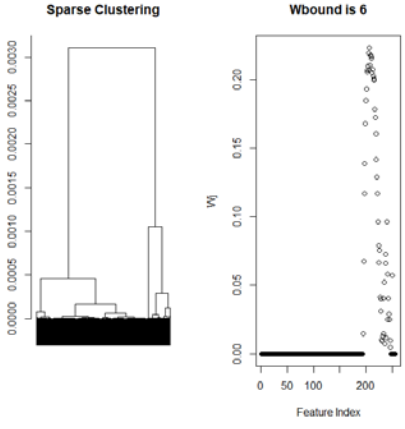
Bu işlemler ardından en başarılı olan modeli çağırabiliriz.

```
sparsehc <- HierarchicalSparseCluster(dists=perm.out$dists,
wbound=perm.out$bestw, method="complete")
```

Şekil 22. En başarılı gelen ağırlık katsayısı ve sonuçları

```
Wbound is  6 :
Number of non-zero weights:  54
Sum of weights:  5.999156
```

Şekil 23. Değişkenlerin Ağırlık Katsayıları



7. SONUÇ

Yapılan çalışmalar ve uygulama örnekleri sonucunda seyrek kümeleme yönteminin klasik kümeleme yöntemlerine göre zaman zaman daha iyi zaman zaman ise benzer performasn gösterdiği görülmüştür. Seyrek kümeleme ile beraber değişken katsayıları üzerinde oynanması ve gerekli olan optimizelerin yapılması daha sağlıklı bir kümeleme çalışması yapılmasına olanak sağlamaktadır. Kullanım kolaylığı ve erişilebilir olması ile beraber özellikle gözlem sayısının değişken sayısından çok küçük olduğu durumlarda oldukça kullanışlı bir yöntem olduğu görülmüştür.

KAYNAKÇA

Hastie, T., Tibshirani, R., & Wainwright, M. (2015). Statistical Learning with Sparsity: The Lasso and Generalizations (1st ed.). Chapman and Hall/CRC.

https://cran.r-project.org/web/packages/sparcl/index.html

https://rdrr.io/cran/rrcov/man/fruit.html

https://rdrr.io/cran/sparcl/

K. Krishna and M. Narasimha Murty, "Genetic K-means algorithm," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 29, no. 3, pp. 433-439, June 1999, doi: 10.1109/3477.764879.

Kondo, Y., Salibian-Barrera, M., & Zamar, R. (2016). RSKC: An R Package for a Robust and Sparse K-Means Clustering Algorithm. Journal of Statistical Software, 72(5), 1–26.

Murtagh, F., & Contreras Albornoz, P. (2011). Algorithms for hierarchical clustering: An Overview. In Data Mining and Knowledge Discovery, Wiley Interdisciplinary Reviews (WIRES) (Wiley Interdisciplinary Reviews). John Wiley & Sons.

Witten, D. M., & Tibshirani, R. (2010). A Framework for Feature Selection in Clustering. Journal of the American Statistical Association, 105(490), 713–726.

İSTATİSTİKTE TEMEL ROBUSTLIK KAVRAMLARI

F. Sevinç KURNAZ[1]

1. GİRİŞ

Robustlık kavramının bağlı olduğu ve temelini oluşturan bazı yöntemler bulunmaktadır. Keşif amaçlı robust yöntemler, kutu grafiği gibi grafiksel temsillerin, robust bir şekilde tahmin edilen birkaç parametreye (konum, dağılım) dayandığını ve dolayısıyla aykırı değerlerin hakim olmadığı çok net bir veri temsiliyle sonuçlandığını göstermiştir. Robust tahminciler model ihlallerine veya aykırı değerlerin varlığına karşı kararlı olmalıdır. Peki "kararlı" ne anlama geliyor? Konum tahmini örneğini kullanarak bu soruyu yanıtlamak için önemli biçimsel kavramlar üzerinde durulacaktır.

Robust istatistik yöntemleri tahmin etmede, hipotez testlerinde ve regresyon modellerinde kullanılabilmektedir. Robust yöntemlerde aranan en önemli özellikler, yüksek kırılma noktasına (high breakdown point) ve yüksek etki fonksiyonuna (influence function) sahip olmasıdır.

Bir tahmin edicinin kırılma noktası, genel (global) robustlık ölçüsü olarak tanımlanır ve tahmin ediciyi bozan en küçük sayıdaki aykırın değerlerin, örneklem sayısına oranı olarak verilir.

Örneklem ortalamasının kırılma noktasının $1/n$ olduğu ve $n \to \infty$ için sıfır olduğu bilinmektedir. Yani sadece bir tane

[1] Doç. Dr., Yıldız Teknik Üniversitesi Fen-Edebiyat Fakültesi İstatistik Bölümü, fskurnaz@yildiz.edu.tr, ORCID: 0000-0002-5958-7366.

aykırı değer olması bile örneklem ortalamasının bozulmasına yetecektir. Diğer taraftan örneklem medyanının kırılma noktası %50'dir. Bu durumda, verilerin yarısına yakını bozuk olsa bile örneklem medyanı tutarlı bir tahmin edicidir.

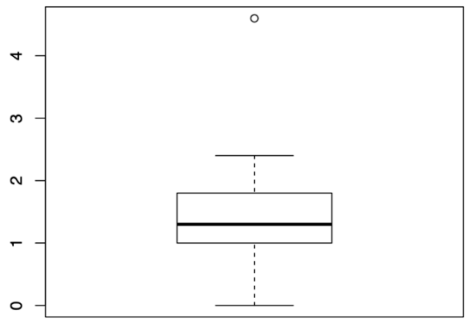
Etki fonksiyonu ise genel olarak yerel dayanıklılığın ölçüsüdür ve bir tek gözlemin çok büyük veya çok küçük olması durumunun tahmin ediciye etkisini ölçer.

Etki fonksiyonunun dönüştürülmesi ile elde edilen duyarlılık eğrisi (sensitivity curve), mevcut veri setine yeni bir gözlem eklendiğinde tahmin edicide ortaya çıkabilecek maksimum etkiyi ölçmektedir.

Örnek: Chushny ve Peebles tarafından kullanılmış olan veri kümesini ele alalım (Chushny ve Peebles, 1905). Bu veri kümesinde, iki tane uyku ilacının uyku süresini veren $n = 10$ tane gözlem dikkate alınmıştır. Bu veri kümesi iki ilacın gözlenen etkisi arasında önemli bir fark olup olmadığını araştırmak için t-test sonucuna bakmak üzere bir öğrenci tarafından kullanılmıştır. Veri kümesine ait kutu grafiğine (boxplot) bakalım.

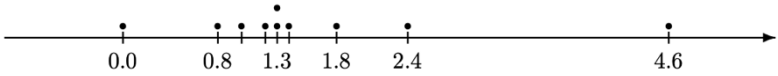
```
> xdat <- c(0.0,0.8,1.0,1.2,1.3,1.3,1.4,1.8,2.4,4.6)
> boxplot(xdat)
```

Şekil 1. Chushny ve Peebles Verisi İçin Kutu Grafiği



Kutu grafiği özellikle büyük veri kümelerindeki aykırı değerlerin belirlenmesinde kullanılan bir yöntemdir. R içerisindeki t.test() fonksiyonu, ortalama için güven aralığını belirlemek veya yokluk hipotezini test etmek için kullanılabilmektedir. Ancak, Şekil 1'deki kutu grafiğinden anlaşılacağı üzere, bu veri kümesindeki 4.6 değeri aykırı değer olarak belirlenmektedir. Şekil 2'de doğru üzerindeki verinin duruşu da bu sonucu desteklemektedir.

Şekil 2. Chushny ve Peebles Verisi



Aykırı değer(ler) analiz sonuçlarını çok büyük ölçüde değiştirebilmektedir. Örneğin, eldeki örnekte aykırı değer olarak belirlenen 4.6 gözlemi veri kümesinden çıkarılmadan aritmetik ortalama hesaplandığında sonuç aykırı değer yönüne doğru bir eğilim gösterecektir.

Aykırı değer veri kümesinden çıkarılarak aritmetik ortalama hesaplandığında ise sonuç oldukça farklı olacaktır. Hatta, aykırı değer çıkarıldıktan sonra elde edilen aritmetik ortalamanın konum hakkında iyi ve güvenilir bir tahmin sağlayacağından emin olunabilir.

```
> t.test(xdat)

One Sample t-test

data: xdat

t = 4.0621, df = 9, p-value = 0.002833

alternative hypothesis: true mean is not equal to 0

95 percent confidence interval:

0.7001142  2.4598858
```

```
sample estimates:
mean of x
    1.58
> mean(xdat)
  [1]  1.58
> sd(xdat)
  [1]  1.22995
> t.test(xdat)$conf.int
  [1]  0.7001142   2.4598858
  attr(,"conf.level")
  [1]  0.95
```

Eldeki örneğe göre, 4.6 değeri normal model için bir aykırı değer olarak belirlenmiştir. Bu durumda, konum ve ölçek parametrelerinin robust tahmin edicilerini kullanmak faydalı olacaktır.

Bununla birlikte, aykırı değerler genellikle hatalardan kaynaklanan değerler değil, yalnızca "şans eseri" çok fazla sapan, doğru şekilde ölçülen değerlerdir. Bu tesadüfi sapma, dağılımın makul bir değeri olabilir (örneğin, uzun kuyruklu), ancak aynı zamanda başka etkilerden kaynaklanan başka bir dağılımdan da gelmiş olabilir.

Aykırı değerlerin tespit edilmesi görevi elbette robust istatistiklerin yalnızca bir yönüdür. Ayrıca robustlık hakkında tek boyutlu konumun tahmininden başka söylenecek daha çok şey vardır. Ancak bu basit örnek kullanılarak birçok önemli kavram açıklanabilir.

Peki, verilerde aykırı değerlerden şüpheleniliyorsa konum nasıl tahmin edilebilir:

Aykırı değerleri göz ardı etmek: Konum için yapılan bu tahminin büyük ölçüde çarpıtılması riskini göze alarak hâlâ klasik aritmetik ortalamayı hesaplıyoruz.

Örnek ortalaması:

$$\bar{x} = 1.58$$

Aykırı değerlerin etkisini azaltmak: $\%\alpha$ kırpılmış ortalamayı hesaplıyoruz. Veriler boyuta göre sıralanır ve verilerin ortadaki $(100 - 2\alpha)\%$'sinin ortalaması alınır. α =%10 için örneğimizde 8 gözleme ihtiyacımız var:\\

%10 kırpılmış ortalama:

$$\frac{0.8 + 1.0 + 1.2 + 1.3 + 1.3 + 1.4 + 1.8 + 2.4}{8} = 1.4$$

Aykırı değerlerin etkisini önemli ölçüde azaltmak: Yukarıdaki durumda, eğer α %50'ye yaklaşırsa örnek medyanı elde ederiz. Bu nedenle medyan, çok yüksek aykırı değer yüzdesine kıyasla istikrarlı olmaktadır:

Örnek medyan:

$$\text{medyan= } 1.30$$

Önce aykırı değerleri kaldırmak: Aşağıdaki gibi sağlam bir dağılım ölçüsü hesaplıyoruz:

Mutlak sapmaların medyanı (MAD):

$$\begin{aligned} \text{MAD} &= \text{med}_i|\text{x}_\text{i} - \text{Medyan}| \\ &= \text{med}\{1.3, 0.5, 0.3, 0.1, 0.0, 0.1, 0.5, 1.1, 3.3\} \\ &= 0.4 \end{aligned}$$

Daha önce de belirttiğimiz gibi aykırı değerler, verinin yayılmasına bağlı olarak merkezden uzakta olan değerler olarak anlaşılmaktadır. Artık hem merkez hem de dağılım robust bir şekilde tahmin edilmiştir ve aralığın dışındaki tüm gözlemler

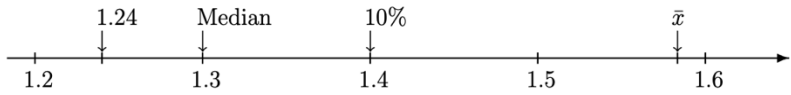
[Medyan-4.MAD, Medyan+4.MAD]=[-0.3,2.9]

aykırı değerler olarak ilan edilebilir ve kaldırılabilir. (Bizim örneğimizde sadece gözlem 4.6 çıkarılmıştır.) Geriye kalanın aritmetik ortalaması hesaplanmıştır.

Aykırı değerlerin olmadığı ortalama:

$$\frac{0.0 + 0.8 + 1.0 + 1.2 + 1.3 + 1.3 + 1.4 + 1.8 + 2.4}{9} = 1.24$$

Konum tahmini için bu farklı seçenekler Şekil 3'te görselleştirilmiştir:



Şekil 3: Chushny ve Peebles verisi

Konum tahminlerinden hangisi "en iyisidir"? Büyüklüklerden hangisi veri dağılımının merkezini en iyi şekilde yansıtır? Robust bir tahmincinin hangi özelliklere sahip olması gerekir? Bu tahminleri (ve diğerlerini) bazı yönlere göre karşılaştırabilmek için doğru araca ihtiyacımız var. Büyük oranda aykırı değerlere veya varsayımlardan sapmalara karşı dayanıklı olması beklenmektedir. Bunu ölçmek için çeşitli robustlık ölçütü kavramlarının açıklanması gerekmektedir.

2. ROBUSTLIK ÖLÇÜTLERİ

Bu bölümde literatürde kullanılan çeşitli robustlık ölçütleri (Huber, 1981; Maronna, 2006) ayrıntılı şekilde ele alınacaktır.

2.1. Hassasiyet Eğrisi

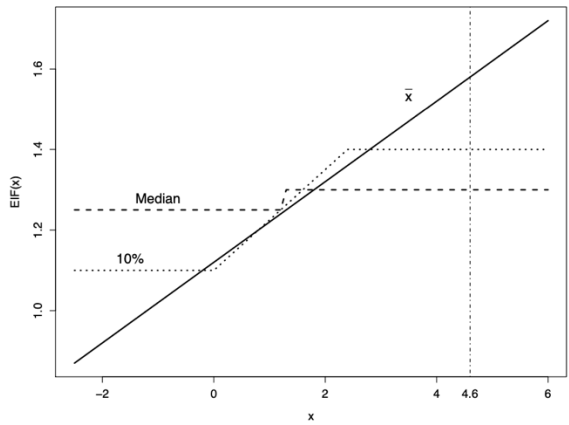
Bir gözlemin farklı konum tahmini olasılıkları üzerindeki etkisini daha görünür kılmak için aşağıdaki deneyi gerçekleştiriyoruz:

Önceki örnekteki en büyük gözlem olan 4.6'nın $-\infty$ ile ∞ arasında değişmesine izin verirsek tahminde ne olur?

Bu deney, bu gözlemin diğer değerlerinin konum tahmini üzerinde ne gibi bir etkiye sahip olacağını görmemizi sağlar. Bu şekilde, özel bir konum tahmincisi T kullanılarak bu veri seti için bir tür ampirik etki fonksiyonu (EIF) oluşturulur.

$$EIF(x) = \mathrm{T}(0.0, 0.8, \ldots, 2.4, \mathrm{x})$$

Şekil 4 aritmetik ortalama $\bar{x}$, %10 düzeltilmiş ortalama ve medyan için EIF'yi göstermektedir.



Şekil 4. Cunshny ve Peebles'den (1905) elde edilen değiştirilmiş veriler için, konum tahmincilerinin aritmetik ortalaması $\bar{x}$, %10 kesilmiş ortalama ve medyan ile ampirik etki fonksiyonu

Şekil 4'den aritmetik ortalama için EIF'nin sınırsız olduğu, hem %10'luk kırpılmış ortalamanın hem de medyanın sınırlı bir EIF'ye sahip olduğu hemen görülmektedir. Bu nedenle tek bir aykırı değerin $\bar{x}$, tahmini üzerinde keyfi olarak güçlü bir etkisi vardır.

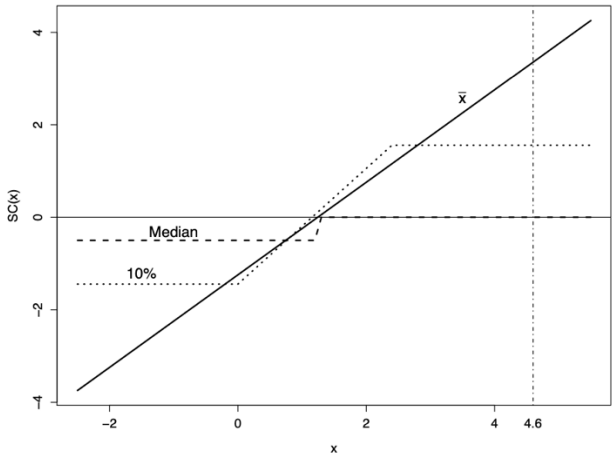
Şimdi EIF basitleştirilecek:

- merkezileştirme (“saf” tahmin çıkarılarak)
- standartlaştırma (kirlilik oranı 1/n'e bölünerek)

Böylece aşağıda verilen hassasiyet eğrisi (sensitivity curve - SC) formülü elde edilir.

$$SC(x) = \frac{T(0.0,\ldots,2.4,x) - T(0.0,\ldots,2.4)}{\frac{1}{10}}$$

Bu elbette SC'nin aritmetik ortalama $\bar{x}$, ile sınırsız olduğu anlamına gelir (bkz. Şekil 5).



Şekil 5: Konum tahmincilerinin aritmetik ortalaması $\bar{x}$, %10 kesilmiş ortalama ve medyan ile Cunshny ve Peebles'den (1905) değiştirilmiş verilere yönelik hassasiyet eğrisi

2.2. Etki Fonksiyonu

Ampirik dağılım fonksiyonu $F_n = \frac{1}{n}\sum_{i=1}^{n} I_{(-\infty,x]}(x_i)$, örneklem büyüklüğü n arttıkça popülasyonun temel dağılımı F'e yakınsar. Hassasiyet eğrisinin de bir sınırı vardır:

T bir tahminci ve x bir "aykırı değer" olsun. IF, uzaydaki konumuna göre, sonsuz küçük miktardaki bir kirliliğin bir tahminci üzerindeki etkisini ölçer. Daha açık olarak, bir T tahmin edicisinin belirli bir F dağılımındaki etki fonksiyonu şu şekilde tanımlanır:

$$IF(x;T,F) = \lim_{n\to\infty} \frac{T(1-\frac{1}{n}F + \frac{1}{n}\Delta_x) - T(F)}{\frac{1}{n}}$$

Burada 1/n kirlenme oranını, Δ_x ise x noktasındaki toplam kütle ile olasılık dağılımını belirler.

Etki fonksiyonu bu nedenle T tahmini üzerindeki sonsuz küçük bir kirlenmenin etkisini ölçer. IF'nin aynı zamanda spesifik dağılım fonksiyonu F'ye de bağlıdır.

Yeterince büyük n için,

$$T(x_1, x_2, \ldots, x_{n-m}, x, x, \ldots, x) \approx T(x_1, x_2, \ldots, x_{n-m}) - \frac{m}{n} IF(x;T,F)$$

Sadece konum tahmincileri olması gerekmez, IF(x;T,F) çoğu T tahmincisi için bulunabilir.

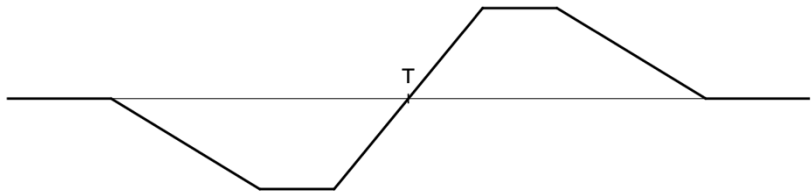
Daha önceki bölümde EIF ve SC ile belirttiğimiz gibi sınırlı IF'ye sahip tahmin ediciler sağlamlık açısından tercih edilmelidir. IF sınırına brüt hata duyarlılığı adı verilir ve şu şekilde tanımlanır:

$$\gamma^*(T,F) = sup_x|IF(x;T,F)|$$

Tahminci üzerindeki sonsuz küçük kirlenmenin en büyük etkisini ölçer. Eğer $\gamma^*(T,F)$ sonlu ise, F üzerinde B-sağlam yanlılığına sahip bir tahmin edici olan T'den söz edilir.

Örnek: Aritmetik ortalama için IF'nin tam biçimini bilmeden, önceki bölümde bu durumda IF'nin sınırsız olması gerektiğini zaten görmüştük. Bu nedenle, örneğin F normal dağılım ise

$\gamma^*(\bar{x}, F = \infty)$ olur. Kırpılmış ortalama veya medyan için γ'nın hesaplanması yalnızca IF bilgisi ile mümkündür. Bir yandan, bilinen tahminciler için etki fonksiyonu hesaplanabilir, ancak bunun tersi bir yaklaşım da düşünülebilir: IF'nin "istenen" bir formu belirlenebilir ve böylece yeni bir tahminci oluşturulabilir. Cushny ve Peebles (1905) verileriyle konum tahmini için aşağıdaki IF formu ele alınabilir.



Bu nedenle dağılımın T merkezine yakın gözlemlerin etkisi doğrusaldır (aritmetik ortalamada olduğu gibi). T'den belirli bir mesafede etki sınırlıdır (kesik ortalamada olduğu gibi). Daha uzaktaki gözlemler için etki o zaman daha da azalır ve sonuçta merkezden aşırı derecede uzakta olan gözlemlerin artık T konum tahmini üzerinde herhangi bir etkisi olmaz. Bu, etkili gözlemlerden artık hiçbir etkisi olmayanlara doğru "sürekli" bir geçiş yaratır.

Böylece artık bir tahmincinin oluşturulacağı belirli bir şekle sahip bir IF verdik. $\psi(x)$ IF'nin istenilen formu olsun. Daha sonra M-tahmincisi (maksimum olabilirlik için "M") kapalı formdaki denklemle tanımlanır:

$$\sum_{i=1}^{n} \psi\left(\frac{x_i - T}{MAD}\right) = 0$$

İteratif algoritmalar genellikle konum tahmincisi T'yi hesaplamak için kullanılır.

2.3. Yerel Değişim Hassasiyeti

Etki fonksiyonu, sonsuz küçük kirlenmenin, yani verinin küçük bir kısmının bozulmasının, ancak keyfi olarak konumlandırılmasının etkisinin bir tahmincisini karakterize eder. Ancak bir tahmincinin yerel karışıklıklar durumunda nasıl davrandığı da ilginç bir konudur. Peki bir gözlemi "uydurursanız" ne olur? Bu tür etkiler örneğin yuvarlama, gruplama veya o andaki yerel yanlışlıklardan kaynaklanabilir.

IF'de belirgin sıçramalar meydana gelirse, bu, küçük bir dalgalanmanın bile tahminde ani bir değişikliğe neden olabileceği anlamına gelir. Bu nedenle sürekli bir IF ile ilgilenilir ve IF'nin türevinin mevcut olduğu yerde sınırlı olması gerekmektedir.

Bir x gözlemini yakındaki bir gözleme taşımanın etkisi ortaya çıkan gözlem şu şekilde ölçülebilir:

$$IF(y;T,F) - IF(x;T,F)$$

Eğer y–x mesafesiyle normalleştirme yaparsanız, elde edilen boyut x ve y noktaları arasındaki IF artışına karşılık gelir. En büyük artış yerel kayma hassasiyeti olarak tanımlanır ve şu şekilde tanımlanır:

$$\lambda^* = sup_{x \neq y} \frac{|IF(y;T,F) - IF(x;T,F|}{|y - x|}$$

Örneğin, aritmetik ortalamanın yerel değişim hassasiyeti $\lambda^* = 1$, $\%\alpha$ kırpılmış ortalamanın yerel değişim hassasiyeti $\lambda^* = \frac{1}{1-\alpha}$, medyanın yerel değişim hassasiyeti $\lambda^* = \infty'$dir.

2.4. Maksimum Yanlılık Eğrisi

Verideki küçük miktarlarda kirlenmenin tahmin edici üzerindeki etkisi IF ile ölçülmektedir. Eğer, verideki kirlenme büyük boyutlarda olursa ne olur? Robust tahmin edicilerden beklenti, belirli bir kirlenme oranına dayanıklı olması yönündedir. Verilerdeki kirlenme oranına göre bir tahmincinin ne

ölçüde bozulduğunu incelemek için kullanılan matematiksel araç, maksimum yanlılık eğrisidir. Maksimum yanlılık eğrisi, bir tahmincinin olası en kötü kirlenme türünün yüzdesine göre sahip olduğu direnci ölçer. X orijinal veri seti olsun ve Y ise n gözlemden m'sinin keyfi değerlerle değiştirildiği bir veri seti olsun. Buna göre, bir T tahmincisi için maksimum yanlılık eğrisi şu şekilde tanımlanır:

$$maxbias(m, T, X) = \sup_Y \| T(Y) - T(X) \|$$

Bazı regresyon tahminleri için mümkün olan en kötü aykırı değer türünün, y, x ve y/x oranının sonsuza doğru artış gösterdiği noktalarda bulunduğu bilinmektedir. En küçük kareler regresyon tahmincisi gibi robust olmayan tahmin edicilerin, zaten küçük miktarlarda kirlenme durumunda maksimum sonsuzluğa ulaşmaktadır. Bu durum ise bizi daha önemli bir kavramı olan Kırılma Noktası'nı düşünmeye itmektedir.

2.5. Kırılma Noktası

IF, bir tahmincinin küçük bir aykırı değerler oranına nasıl tepki verdiğini gösterir. Aritmetik ortalamanın tek bir aykırı değere karşı bile robust değildir (diğer sınırlı IF tahmincilerinin aksine).

Aykırı değerlerin sayısının tahmincilerin (sınırlı IF ile) artık bu etkiye karşı dirençli olmayacak kadar büyük olması mümkün müdür?

Bu soru bizi tahmincilerin yeni bir karakterizasyonuna, yani tahmincinin kırılma noktası (Breakdown Point - BP) olarak adlandırılan noktaya götürür. BP, tahmincinin keyfi değerler almasına neden olabilecek en küçük kirlilik miktarıdır.

Tanım: Maksimum yanlılık eğrisinden faydalanarak, verilen bir X verisi için T tahmin edicisinin kırılma noktası

$$\varepsilon_n^* = \min \left\{ \frac{m}{n}; maxbias(m, T, X) = \infty \right\}$$

şeklinde tanımlanır.

$n \to \infty$ için ε^* ile gösterilen asimptotik kırılma noktası elde edilir. Yani, $\lim_{n\to\infty} \varepsilon_n^* = \varepsilon^*$'dir.

Örneğin, aritmetik ortalamanın kırılma noktası $\varepsilon^* = \%0$, $\%\alpha$ kırpılmış ortalamanın kırılma noktası $\varepsilon^* = \%\alpha$, medyanın kırılma noktası ise $\varepsilon^* = \%50$' dir.

Kırılma noktasının alabileceği maksimum değer %50'dir. Daha büyük bir değer olamaz, çünkü bu durumda aykırı değerlerin çoğunluğu "iyi" veri değerleri olarak görülebilir.

KAYNAKÇA

Huber, P. J. Robust Statistics, Wiley, 1981.

Maronna, R. A., Martin, R. D. and Yohai, V. J. Robust Statistics Theory and Methods. John Wiley and Sons Ltd.: England., 2006.

Peebles A.R. and Chushny, A.R. The action of optical isomers. II. Hyoscines, The Journal of Physiology, 32(5-6), 1905.

KONUM VE ÖLÇEK PARAMETRELERİ İÇİN BAZI TAHMİN EDİCİLER

F. Sevinç KURNAZ[1]

1. GİRİŞ

Konum parametresi (location parameter) ve ölçek parametresi (scale parameter) için öncelikle aşağıda verilen modeli düşünelim:

$$x\_i = \mu + \sigma\varepsilon_i, \qquad i = 1,2,\dots,n$$

Bu modelde $\mu \in \mathbb{R}$ bilinmeyen yer parametresini, $\sigma > 0$ ölçek parametresini, ε_i ise bilinmeyen bir F dağılımına göre bağımsız ve aynı dağılımlı rastgele değişkenleri göstersin. Eğer bu F dağılımı standart Gauss dağılımı ise; aritmetik ortalama, standart sapma, t-test, vs. gibi klasik çıkarımsal yöntemlerin kullanılması oldukça yaygındır ve güzel sonuç verir.

Verilen bir $x = \{x_1, x_2, \dots, x_n\}$ örneklemi için σ bilindiğinde konum parametresi μ için bir tahmin edici bulmak ile ilgilenmekteyiz. Böyle bir veri için tahmin edici (EKK) olarak örneklem ortalaması kullanılması oldukça yaygındır. Benzer şekilde ölçek parametresi için kullanılan tahmin edici ise örneklem standart sapmasıdır. Ancak bilinmektedir ki, bu tahmin ediciler robust olmadıklarından, veri kümesinde aykırı değerler olduğunda yanıltıcı sonuç vermektedirler.

[1] Doç. Dr., Yıldız Teknik Üniversitesi Fen-Edebiyat Fakültesi İstatistik Bölümü, fskurnaz@yildiz.edu.tr, ORCID: 0000-0002-5958-7366.

2. ÖRNEKLEM ORTALAMASI

Örneklem ortalaması

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Örneklem ortalaması bir konum ölçüsüdür ve evren ortalamasının bir tahmin edicisidir. Örneklem ortalaması aykırı değerlerden çok fazla etkilenebilmektedir, bu nedenle aykırı değerlere karşı robust değildir.

3. ÖRNEKLEM MEDYANI

Örneklem medyanı, bir veri setinde sadece ortadaki bir veya iki gözlemin farklı ağırlıklandırıldığı, diğer tüm gözlemlerine sıfır ağırlığının verilmesi ile tanımlanmaktadır. Matematiksel olarak ifade etmek istersek, örneklem medyanı

$$\text{Medyan} = \begin{cases} x_{\frac{n}{2}} & n\ tek \\ \dfrac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2} & n\ çift \end{cases}$$

şeklinde tanımlanmaktadır.

Örneklem medyanı verilerin merkezindeki değere eşit olup, sapan değerlerden etkilenmemektedir, dolayısıyla robust bir tahmin edicidir.

4. ÖRNEKLEM VARYANSI

Örneklem varyansı

$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$

olarak verilmektedir. Örneklem varyansı örneklem ortalaması kullanılarak hesaplanmakta olur, sapan değerlere karşı oldukça hassastır ve robust olmayan bir ölçek tahminidir.

5. ÖRNEKLEM İÇİN MEDYAN MUTLAK SAPMASI

Örneklem varyansına robust bir alternatif, örneklem medyanının kullanılması ile elde edilen Medyan Mutlak Sapması (Median Absolute Deviations - MAD) olarak adlandırılmaktadır. Medyan mutlak sapması

$$\text{MAD} = \text{medyan}|x_i - \text{medyan}(x_i)|$$

ile verilmektedir.

6. KIRPILMIŞ ORTALAMA

Aykırı değer içeren bir veri kümesi üzerinde kırpılmış ortalamayı hesaplamak bir robust bir yöntem olarak ele alınabilir. Kırpılmış ortalama dağılımın $1 - 2\alpha$'lik merkezinin ortalamasıdır. Böylece uç noktalardaki αn gözlemler kırpılır.

7. M TAHMİN EDİCİLERİ

Gauss modelleri ile çalışıldığından, medyan ve MAD robust tahmin ediciler olsalar da etkili tahmin ediciler değillerdir. Bunun yerine hem robustlık hem de etkinlik özelliklerinin bir kombinasyonuna sahip olan M tahmin edicilerini kullanmak daha makuldür. Eğer σ biliniyorsa, μ'nün bir M-tahmin edicisi, aşağıdaki denkleminin bir çözümü olarak olarak elde edilir:

$$\sum_{i=1}^{n} \psi_k \left(\frac{x_i - \mu}{\sigma} \right) = 0$$

Burada $\psi_k()$ için literatürde çeşitli versiyonlar seçilebilmektedir. Eğer ölçek parametresi σ bilinmiyorsa, bu durumda aşağıdaki eşitliğin çözülmesiyle bilinmeyen ölçek parametresini tahmin etmek mümkündür.

$$\frac{1}{(n-1)}\sum_{i=1}^{n}\chi\left(\frac{x_i-\mu}{\sigma}\right)=k_2$$

Alternatif olarak, σ ölçek parametresi $\hat{\sigma}_{MAD}$ ile tahmin edilebilir. Bazı simülasyon sonuçları başlangıç ölçek tahmin edicisi olarak ele alınan MAD tahmin edicisinin kullanılmasının konum parametresi için M tahmin edicilerinin üstünlüğüne sebep olduğunu göstermiştir.

Huber fonksiyonu konum parametresi için ölçek parametresini MAD ile tahmin ederek Huberin M tahmin edicisini bulur (Huber 1981; Venables and Ripley 2002). Bu durumda $\psi_k()$ fonksiyonu

$$\psi_k(x)=max(-k,x)$$

şeklindedir. Burada k sabiti kullanıcı tarafından belirlenir. Tahmin edicinin etkinliğinin ve robustlığının arasındaki dengeyi sağlayacak şekilde belirlenmesi gerekmektedir. $k\to 0$ iken medyanı, $k\to\infty$ iken aritmetik ortalamayı vermektedir. Gaussian modelde $k=1.345$ değeri, %95 etkin bir tahmin ediciyi verir.

8. R FONKSİYONLARI İLE UYGULAMALAR

8.1. Mean() Fonksiyonu ile Uygulama

Kırpılmış ortalamayı hesaplamak için α kırpılma seviyesi ile birlikte mean() fonksiyonu kullanılır.

```
mean(x, trim=0)
```

Burada x bir vektör veya dataframe olacak şekilde tanımlı olabilir. trim ise x datasının her bir ucundan kesilecek gözlemlerin oranını (0 ile 0.5 arası) göstermektedir. Eğer trim, 0 varsayılan değerini alıyorsa, klasik aritmetik ortamala hesaplanır. Eğer trim sıfırdan farklı bir değer alıyorsa, aldığı değer kadar kırpılmış ortalama hesaplanır.

Örnek - Chushny and Peebles Verisi

İki ilaç yardımıyla uyku süresinin uzatılmasına ilişkin $n = 10$ gözlemden oluşan veriyi ele alalım.

```
> xdat
[1] 0.0 0.8 1.0 1.2 1.3 1.3 1.4 1.8 2.4 4.6
> mean(xdat)
1.58
> mean(xdat,trim=0.1)
1.4
> mean(xdat,trim=0.2)
1.333333
> mean(xdat[1:9])
1.244444
```

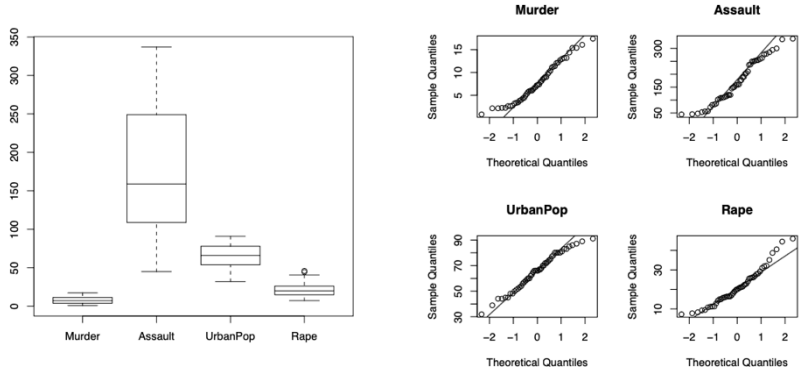
Buradaki sonuçlara bakacak olursak, kırpılmış ortalama (robust tahmin edici), klasik artimetik ortalamadan ve sonuncu gözlem 4.6 değeri veriden çıkarılarak hesaplanan tahmin sonucundan oldukça farklıdır.

Örnek - USArrests Verisi

USArrests veri seti, 4 değişken ve 50 gözlemden oluşmaktadır. 1973 yılında ABD'nin 50 eyaletinin her birinde saldırı (assault), cinayet (murder), tecavüz (rape) nedeniyle her 100.000 kişiye

düşen tutuklama sayılarını vermektedir. Ayrıca verilen nüfusun kentsel bölgelerde yaşayanlarının yüzdesini (UrbanPop) de verilmektedir. Bu verimkümesine ait kutu grafiği (boxplot) Şekil 1 (a)'da görülebilir. Klasik aritmetik ortalama, %20'lik kırpılmış ortalama vs. hesaplanabilir. Örneğin %20'lik kırpılmış ortalamayı hesaplarken en küçük ve en büyük %20'lik gözlemler kırpılarak ortalama hesaplanır. Şöyle ki:

Şekil 1. USA Arrests Verisi için Kutu Grafiği (a) ve QQ-Normal Grafiği (b)



(a) (b)

```
> data(USArrests
> boxplot(USArrests)
> mean(USArrests)
Murder    Assault   UrbanPop      Rape
 7.788    170.760     65.540    21.232
> mean(USArrests, trim=0.2)
Murder    Assault   UrbanPop      Rape
 7.42     167.60      66.20     20.16
```

Buradaki sonuçlara göre, kırpılmış ortalama ile klasik aritmetik ortalama arasında büyük bir farklılık görülmemektedir. Bu durumu genellemek doğru olmaz. Şekil 1 (b)'deki Normal QQ grafiklerinden veride aykırı değer olduğu bu durumda, Gauss dağılımından küçük bir sapmaya göre robust olan tahmin edicinin tercih edilebileceği bilgisi çıkarılabilir.

```
> par(mfrow=c(2,2))
> qqnorm(USArrests\$Murder,main="Murder")
> qqline(USArrests\$Murder)
> qqnorm(USArrests\$Assault,main="Assault")
> qqline(USArrests\$Assault)
> qqnorm(USArrests\$UrbanPop,main="UrbanPop")
> qqline(USArrests\$UrbanPop)
> qqnorm(USArrests\$Rape,main="Rape")
> qqline(USArrests\$Rape)
```

8.2. Median() ve Mad() Fonksiyonları ile Uygulama

Konum ve ölçek parametrelerinin iki basit robust tahmin edicisi sırasıyla medyan ve MAD (medyan mutlak sapma) dir. median() ve mad() fonksiyonları kullanılarak hesaplanabilirler. Burada median() fonksiyonu klasik olarak verilen bir x örneklemi için örneklem medyanını hesaplar. Medyan robust bir tahmin edicidir ve %50'ye kadar veride aykırı değer olması durumuna karşı robusttır. Yani, medyan tahmin edicisinin $\hat{\mu}_{me}$ kırılma noktası 0.5'dir. Öte taraftan klasik örneklem aritmetik ortalamasının kırılma noktası 0'dır. median fonksiyonunun kullanımı aşağıdaki gibidir:

```
median(x, na.rm=FALSE)
```

na.rm argümanı, hesaplama devam etmeden önce kayıp değerlerin (NA) çıkarılması gerekip gerekmediğini gösteren mantıksal bir değerdir.

Birçok uygulamada ölçek parametresi çoğunlukla bilinmez ve tahmin edilmesi gerekir. Çeyrekler arası aralık ve MAD gibi σ'nın birkaç robust tahmincisi vardır. Daha basit fakat daha az robust olan bir ölçek tahmin edicisi çeyreklerarası aralık (interquartile range) tahmin edicisidir. Bu tahmin edicis IQR fonksiyonuyla hesaplanabilir. Diğer bir robust ölçek tahmin edici ise $\hat{\sigma}_{MAD} = k\, med(|x_i - \hat{\mu}_{me}|)$ şeklindedir. Burada $\hat{\mu}_{me}$ i. residüeli göstermektedir.

mad() fonksiyonu MAD tahmin edicisini hesaplar ve (varsayılan olarak) bir k sabiti kullanılarak tahmin edicisinin tutarlılığı ayarlanır. mad() fonksiyonunun kullanımı şu şekildedir:

mad(x, center = median(x), constant = 1.4826, na.rm = FALSE)

Bu fonksiyonun içindeki argümalar aşağıdaki gibidir:

- x nümerik bir vektör
- center opsiyonel bir argümandır. Konum parametresi için bir değer alır. (Varsayılan örneklem medyanıdır. Ancak eğer konum parametresi biliniyorsa, farklı bir değer de alabilir.)
- constant ölçek parameteresini ayarlama sabiti k değerini gösterir. (Varsayılan değer $k = 1 - \phi(3/4) \cong 1.4826$, Fisher sabiti)
- na.rm argümanı, hesaplama devam etmeden önce kayıp değerlerin (NA) çıkarılması gerekip gerekmediğini gösteren mantıksal bir değerdir.

$\hat{\mu}_{me}$'nin varyansı $\hat{V}(\hat{\mu}_{me}) = (\pi/2)(\hat{\sigma}^2_{MAD}/n)$ ile tahmin edilir. μ için yaklaşık bir güven aralığı ise $\left(\hat{\mu}_{me} \pm\right.$

$z_{1-\alpha/2}\sqrt{\hat{V}(\hat{\mu}_{me})}\Big)$ şeklindedir. Burada $z_{1-\alpha/2}$ ise standart normal dağılımın $(1-\alpha/2)$ kantilidir.

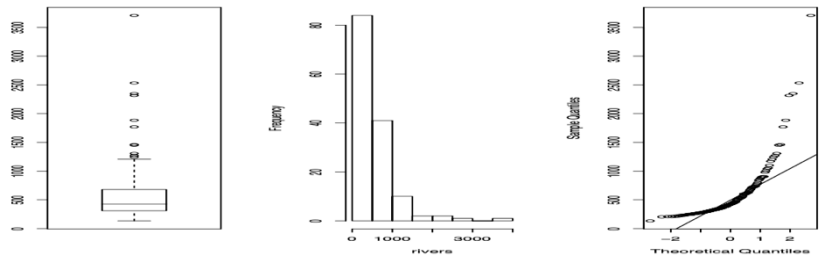
Örnek - Rivers Verisi (McNeil, 1977)

rivers veri seti, ABD Jeoloji Araştırması Kuzey Amerika'daki n = 141 nehrin uzunluğunu (mil cinsinden) vermektedir. Grafikler ve standart konum tahminleri aşağıdaki fonksiyonlar kullanılarak elde edilebilir:

```
> data(rivers)
> par(mfrow=c(1,3))
> boxplot(rivers)
> hist(rivers)
> qqnorm(rivers)
> qqline(rivers)
> median(rivers)
[1] 425
> summary(rivers)
```

Min.	1st Qu.	Median.	Mean	3rd Qu.	Max.
135.0	310.0	425.0	591.2	680.0	3710.0

Şekil 2. Rivers Verisine Ait Grafikler



Şekil 2'de verinin dağılımının simetrik olmadığı görülmektedir. Bu duurmda klasik tahmin edicilerdense robust tahmin edicilerin kullanılması daha uygun olabilir sonucuna varılır. Dolayısıyla, robust ölçek tahmin edicisi aşağıdaki gibi hesaplanır:

```
> mad(rivers)
[1] 214.977
> IQR(rivers)
[1] 370
```

μ için yaklaşık 0,95 güven aralığı hesaplanabilir ve örnek ortalamasına dayalı standart güven aralığından oldukça farklıdır. Şöyle ki:

```
> se.med <- sqrt((pi/2)*(mad(rivers)^ / length(rivers)))
> median(rivers)-qnorm(0.975)}*se.med
[1] 380.5276
> median(rivers)+qnorm(0.975)}*se.med
[1] 469.4724
> t.test(rivers)$conf.int
[1] 508.9559     673.4129
attr(,"conf.level")
[1] 0.95
```

8.3. Huber() ve Hubers() Fonksiyonları ile Uygulama

R programlama dilinde huber() fonksiyonunu kullanmak için MASS kütüphanesinin yüklenmesi gerekmektedir.

```
> library(MASS)
```

komutu ile kütüphane yüklenir. huber() fonksiyonunun kullanımı ise aşağıdaki gibidir:

huber(y,k=1.5)

Bu fonksiyondaki argümanlar

- y verileri içeren vektör
- k Huber'in tahmin edicisinin hesaplanmasında kullanılan sabit

Ölçek parametresi σ'nın değeri bilindiğinde hubers() fonksiyonu kullanılır. hubers() fonksiyonu belirlenen bir σ değeri için konum parametresinin alternatif Huber M tahmin edicisini hesaplar.

hubers(y, k = 1.5, mu, s, initmu = median(y))

hubers() fonksiyonunun argümanları aşağıdaki gibi açıklanabilir:

- y verileri içeren vektör
- k Huber'in tahmin edicisinin hesaplanmasında kullanılan sabit
- mu konum parametresi için belirlenen değer
- s ölçek parametresi için belirlenen değer
- initmu konum parametresi için belirlenen başlangıç değeri

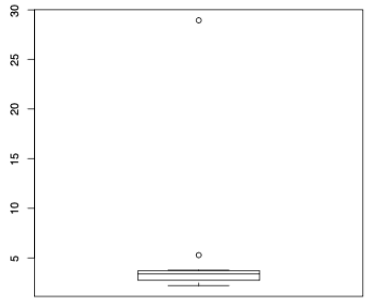
huber() ve hubers() fonksiyonlarının değerleri konum ve ölçek parametrelerinin tahminlerini çıktı olarak verir.

- mu konum tahmini
- s ölçek parametresi için MAD tahmin edicisi ya da Huber'in alternatif ölçek tahmin edicisi

Örnek - chem Verisi

chem verisi, kepekli undaki bakırın milyon başına parça cinsinden tespitini içeren gözlem sayısı $n = 24$ olan bir veri kümesini göstermektedir. Veriye ait kutu grafiği Şekil 3'de verilmiştir ve bu figürde tek bir gözlemin nasıl bir değişim yattığını açıkça görebiliyoruz. Şöyle ki; 28,95 değeri açıkça bir aykırı değerdir. Örneklem açıkça simetrik değildir ve bir değeri 10 kat dışarıda görünmektedir. Sonuçlar bunun doğru olduğu göstermektedir.

Şekil 3. Chem Verisine Ait Kutu Grafiği




```
> data(chem)
> summary(chem)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.200   2.775   3.385   4.280   3.700  28.950
> huber(chem)
$mu
[1] 3.206724
$s
[1] 0.526323
> huber(chem,k=1.345)
$mu
```

```
        [1] 3.216252
$s
[1] 0.526323
> huber(chem,k=5)
$mu
[1] 3.322244
$s
[1] 0.526323
> hubers(chem)
$mu
[1] 3.205498
$s
[1] 0.673652
> hubers(chem,k=1.345)
$mu
[1] 3.205
$s
[1] 0.6681223
> hubers(chem,mu=3)
$mu
[1] 3
$s
[1] 0.660182
> hubers(chem,s=1)
$mu
```

```
[1] 3.250001
```

```
$s
```

```
[1] 1
```

KAYNAKÇA

Bellio, R., and Ventura, L. (2005). An introduction to robust estimation with R functions. *Proceedings of 1st international Work*, *1*(5), 1-57.

Huber, P. J. (1981). Robust Statistics, Wiley.

Maronna, R. A., Martin, R. D. and Yohai, V. J. (2006). Robust Statistics Theory and Methods. John Wiley and Sons Ltd.: England.

Venables, W. N. and Ripley, B. D. (2002). Modern Applied Statistics with S. Fourth edition. Springer.

TÜRKİYE'DE EN ÇOK SATILAN OTOMOBİL FİYATININ HEDONİK FİYAT MODELİ VE YAPAY SİNİR AĞLARI İLE ANALİZİ[1]

Enes ÇÖRDÜKÇÜ[2]

Yüksel TERZİ[3]

Mehmet Şirin ATEŞ[4]

1. GİRİŞ

Ürün fiyatları, hem tüketiciler hem de üreticiler için büyük önem taşır. Tüketiciler için, bir malın fiyatı, talebi etkileyen en temel faktörlerden biridir. Tüketicilerin tercihleri, satın alma güçleri ve elde ettikleri fayda da talebi etkileyen diğer faktörlerdir. Bu nedenle, talep, ürün fiyatı ve kalitesinin belirlenmesinde büyük bir rol oynar. Fiyatı etkileyen bir diğer önemli faktör ise maliyettir. Satıcılar, gerçek amacı olan kar elde etmek için maliyeti minimum seviyede tutarak daha fazla kar sağlayabilmek için çeşitli maliyet yöntemleri uygulayabilirler. Bu, sağlıklı bir fiyatlandırma yapılmasına yardımcı olur (Yayar, 2011).

Otomobil, insanların bir yerden başka bir yere ulaşmasını sağlayan, yük ve eşya taşımak için yaşamı kolaylaştıran ve

---

[1] Bu kitap bölümü Türkiye'de En Çok Satılan Otomobil Fiyatının Hedonik Fiyat Modeli ile Analizi konulu tezden üretilmiştir.

[2] Öğretmen Sınıfı (Subay), Milli Savunma Üniversitesi, Ölçme ve Değerlendirme Daire Başkanlığı, Rektörlük, enescordukcu@gmail.com, ORCID: 0009-0000-4904-351X.

[3] Prof. Dr., Ondokuz Mayıs Üniversitesi, Fen Fakültesi, İstatistik, yukselt@omu.edu.tr, ORCID: 0000-0003-4966-8450.

[4] Araş. Gör., Ondokuz Mayıs Üniversitesi, Fen Fakültesi, İstatistik, mehmet.ates@omu.edu.tr, ORCID: 0000-0001-9904-6380.

kaliteli bir yaşam sunan araçları ifade eder. Otoyolların sembolü haline gelen otomobiller, sadece bir ulaşım aracı değil; aynı zamanda insanların kişiliğini yansıtan, kendilerini güvende hissettikleri, günlük yaşamda kaliteli bir yaşam sürmelerini sağlayan ve vazgeçilmez hale gelen bir araçtır.

2. HEDONİK FİYAT MODELİ

Hedonik kelime anlamı ile hazzın mal ve hizmetlerin tüketimi sonrası ortaya çıkması, tatmin duygusu, memnun olma ya da fayda sağlama gibi anlamlarına gelmektedir. Hedonik fiyat ise kişinin memnuniyeti için ödemeyi göze aldığı miktardır. Bu yöntem ile heterojen bir malı oluşturan bütün özelliklerin, o mal fiyatı üzerindeki etkisi ölçülür. Çünkü bir malın fiyatı, içinde barındırdığı karakteristiklerin bir bütününü oluşturur. Hedonik modellerin kullanım alanları konut, araç, arsa, işyeri, cep telefonu, bilgisayar gibi birçok alandadır.

3. YAPAY SİNİR AĞLARI

Yapay sinir ağları, insan beyninden esinlendiği için ona benzer bir yapıya sahiptir; fakat insan beyninin son derece bağlantılı ve karmaşık işleyişi sadece ona özgüdür. Bu yapı, mevcut bilgisayar programlarının veya diğer alanların ulaşamadığı bir noktadadır ve günümüz araştırmaları bile bu kompleks yapıya oldukça uzaktır. Yapay sinir ağlarını oluşturmak için kullanılan yapay nöronlar, insan beynine kıyasla oldukça basit bir yapıya sahiptir. Dolayısıyla, yapay nöronlar, insan beyninin karmaşıklığından uzak olsa da genel olarak yapısal olarak tutarlıdır. Başka bir ifadeyle, yapay sinir ağları, insan beyninin karmaşık ve güçlü yapısını en basit haliyle kopyalamaya çalışmaktadır (Ataseven, 2013).

Yapay Sinir Ağları (YSA), temel olarak, bir örnekler kümesiyle parametrelerin uyarlanabilmesini sağlayan matematiksel bir formül için yazılan bir bilgisayar programı olarak tanımlanabilir. Bu tanım, YSA'yı en basit ve ayrıntısız şekilde ifade eder.

Son dönemde, problem çözme büyük ölçüde bilgisayarlar tarafından gerçekleştirilmektedir. Bilgisayarlar, insan beynine göre çok daha kısa sürede büyük miktardaki veriyi işleyip problemleri çözebilirler. Sayısal konularda ve problemlerde, bilgisayarlar insan beynine göre önemli ölçüde daha hızlı işlem yaparlar. Ancak, insan beyni ses ve görüntü kaydetme, planlama ve öğrenme gibi işlemleri çok hızlı bir şekilde yapabilirken, bilgisayarlar bu tür işlemleri çok fazla zaman ve alan kullanarak, genellikle düşük başarı oranlarıyla gerçekleştirirler. Bu farklılıklar, bilim insanlarını insan beynini modelleyerek çalışan bilgisayar araştırmaları yapmaya yönlendirir.

Bu yaklaşımın bir sonucu olarak, Yapay Sinir Ağları'nın tanımı şu şekilde yapılabilir: "Yapay Sinir Ağları, insan beyninin özelliklerinden olan öğrenme yoluyla yeni bilgiler üretebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri, herhangi bir yardım almadan otomatik olarak gerçekleştirmek amacıyla geliştirilen bilgisayar sistemleridir."

3.1.Yapay Sinir Ağ Modelleri

Yapay sinir ağlarının mimari yapısı veya topolojisi, yapay nöronların birbirleriyle olan bağlantılarının ve verinin akışının gösterimine denir. İki temel topoloji türü vardır: ileri beslemeli ve geri beslemeli.

İleri beslemeli ağlarda, bilgi girdiden çıktıya doğru yalnızca bir yönde akar, bu da döngüsel olmayan bir grafik oluşturur. Bu nedenle, bu tür ağlara "ileri beslemeli sinir ağları" denir.

Öte yandan, geri beslemeli ağlarda, bilgi sadece girdiden çıktıya değil, aynı zamanda çıkıştan girişe veya nöronun kendisine geri döndüğünde yarı döngüsel grafikler oluşur. Bu tür ağlara ise "geri beslemeli sinir ağları (RNN)" adı verilir (Krenker et al., 2011).

3.1.1. İleri Beslemeli Yapay Sinir Ağları

İleri beslemeli ağlarda, veri giriş katmanındaki nöronlardan işlenerek bir sonraki katmandaki nöronlara iletildiği doğrudur. Bu tür ağlarda geri dönüş yoktur, yani bilgi bir önceki katmandan bir sonrakine doğru ilerler. Ağın katman sayısı ve katmanlardaki nöron sayısı, çözülecek probleme ve tasarım tercihlerine bağlı olarak tasarımcı tarafından belirlenir.

Her katmandaki nöronlar, gelen girdi verilerini ağırlıklarla çarparak toplar. Bu toplam değere nöronun sapma değeri (bias) eklenir ve elde edilen sonuç, aktivasyon fonksiyonu tarafından işlenerek bir sonraki katmandaki nöronlara aktarılır. Bu süreç, ağın girdisinden başlayarak çıktısına doğru ilerler ve genellikle her katmandaki nöronların çıktıları, bir sonraki katmanın girdisi olarak kullanılır. Bu şekilde, ağın her katmanı belirli bir özellik setini öğrenir ve daha karmaşık özelliklerin temsil edildiği daha yüksek seviyeli katmanlara ilerlenir.

3.1.2. Geri Beslemeli Sinir Ağları

Geri Beslemeli Sinir Ağları (GBSA), zaman bükülmesine sahip ileri beslemeli yapay sinir ağlarıdır. GBSA'lar, sıralı veya zaman serisi verilerini kullanan ve önceki adımın çıktısını mevcut aşamaya girdi olarak besleyen ağlardır. Bu ağlardaki nöronlar, önceki girdilerden gelen bilgileri kullanarak mevcut girdi ve çıktıyı etkilemelerine izin veren "bellekleri" ile ayırt edilirler. Tipik ileri beslemeli ağlardan farklı olarak, tekrarlayan ağların çıktısı dizideki önceki öğelere bağlıdır (Sarker, 2021).

Tekrarlayan yapay nöronlar, çıktılarını birkaç kez kendilerine iletebilirler. Bu şekilde, özyinelemeli bir işlevin kendisini çağırması gibi, tekrarlayan bir yapay nöronda kendi çıktısını kullanır. Geri döngülerle ilgili herhangi bir sınırlama bulunmaz (Krenker et al, 2011). GBSA'lar, dil işlemede yaygın olarak kullanılır. Ancak, tekrarlar nedeniyle uzun veri dizilerini öğrenmeyi zorlaştıran, kaybolan gradyanlar sorunuyla karşılaşabilirler. Bu sorunu aşmak için Uzun Kısa Süreli Bellek (LSTM), Geçitli Yinelenen Birimler (GRU'lar) gibi farklı GBSA modelleri geliştirilmiştir

Yapay sinir ağlarının temel bileşenlerinin belirlenmesi oldukça önemlidir ve bu bileşenlerin doğru seçimi, Yapay Sinir Ağlarının (YSA) tahmin performansını doğrudan etkiler. YSA'nın temel bileşenleri genellikle şunlardır: ağ mimarisi, öğrenme algoritması ve aktivasyon fonksiyonu.

3.1.3. Öğrenme Algoritması

Yapay sinir ağlarında öğrenme algoritması, ağı eğitmek ve girdi verileri ile beklenen çıktı arasındaki ilişkiyi öğrenmek için kullanılan matematiksel yöntemlerdir. Bu algoritmalar, ağın parametrelerini (ağırlıklar ve biaslar) ayarlayarak istenilen çıktıyı elde etmeye çalışır.

En yaygın kullanılan yapay sinir ağı öğrenme algoritmalarından biri Geriye Doğru Yayılım (Backpropagation) algoritmasıdır. Bu algoritma, ağın çıkışındaki hata ile geriye doğru giderek her katmandaki ağırlık ve bias değerlerini günceller. Hata, gerçek çıktı ile beklenen çıktı arasındaki farktır. Geriye doğru yayılım algoritması, türev zincir kuralını kullanarak her katmandaki hata payını geriye doğru ileterek ağırlık ve bias değerlerini ayarlar.

Başka bir öğrenme algoritması ise Gradyan İnişi (Gradient Descent) algoritmasıdır. Bu algoritma, ağın hatasını azaltmak için gradyan yöntemini kullanır. Ağın parametrelerini

(ağırlıklar ve biaslar) gradyanın tersi yönünde güncelleyerek hatayı azaltır.

Diğer önemli bir öğrenme algoritması Stokastik Gradyan İnişi (SGD) algoritmasıdır. Bu algoritma, her bir eğitim örneği için gradyanı hesaplayarak ağırlık ve bias değerlerini günceller. Bu sayede eğitim verilerinin rastgele örneklerini kullanarak ağı hızlı bir şekilde eğitir ve daha iyi bir genelleme sağlar.

Bu algoritmalar, yapay sinir ağlarını eğitmek için temel yaklaşımları sağlar. Hangi algoritmanın kullanılacağı, belirli bir problem ve ağ mimarisiyle ilişkilidir.

3.1.4. Aktivasyon Fonksiyonu

Yapay sinir ağlarında aktivasyon fonksiyonu, her bir sinir hücresinin çıktısını belirlemek için kullanılan matematiksel bir fonksiyondur. Bu fonksiyon, bir sinir hücresinin girdi sinyallerini toplar ve belirli bir eşik değerini aşan bir çıktı üretir. Aktivasyon fonksiyonu, sinir ağının öğrenme sürecini ve çıktılarını belirleyen önemli bir bileşendir.

Yapay sinir ağlarında kullanılan yaygın aktivasyon fonksiyonları şunlardır:

i-Sigmoid Fonksiyonu: Sigmoid fonksiyonu, genellikle girişleri 0 ile 1 arasında normalleştirmek için kullanılır. Matematiksel olarak, sigmoid fonksiyonu şu şekildedir.

$$\sigma(x) = 1/(1+e^{-x}) \quad \textbf{(5.1)}$$

ii-ReLU (Rectified Linear Unit) Fonksiyonu: ReLU fonksiyonu, giriş negatif ise sıfır, pozitif ise girişin kendisi olarak çıktı üretir. Matematiksel olarak, ReLU fonksiyonu şu şekildedir:

$$f(x)=\max(0,x) \quad \textbf{(5.2)}$$

iii-Tanh (Hiperbolik Tangant) Fonksiyonu: Tanh fonksiyonu, girişleri -1 ile 1 arasında normalleştirmek için kullanılır. Matematiksel olarak, tanh fonksiyonu şu şekildedir:

$$\tanh(x)= (e^{x} - e^{-x})/(e^{x} + e^{-x}) \quad \textbf{(5.3)}$$

iv-Leaky ReLU (Leaky Rectified Linear Unit), yapay sinir ağlarında kullanılan bir aktivasyon fonksiyonudur. ReLU'nun (Rectified Linear Unit) bir genelleştirmesidir ve özellikle negatif girişler için sıfır yerine küçük bir eğimle devam eden bir doğrusal fonksiyondur. Matematiksel olarak, Leaky ReLU fonksiyonu aşağıdaki gibidir:

$$f(x)=\max(\alpha x,x) \quad \textbf{(5.4)}$$

(5.4)'de, α küçük bir pozitif sabittir (genellikle 0.01 gibi küçük bir değer alır). Yani, giriş x negatif ise, Leaky ReLU fonksiyonu αx değerini alır, aksi takdirde giriş değeri olan x değerini alır.

Leaky ReLU, özellikle ReLU'nun negatif girişlere karşı sıfır gradyan üretmesi sorununu ele almak için kullanılır. Bu, aşırı eğitilmiş ağların daha iyi performans göstermesine yardımcı olabilir. Leaky ReLU, genellikle ReLU'nun yerine veya ağın daha derin katmanlarında kullanılır.

4. VERİ SETİ, METODOLOJİ VE DEĞİŞKEN SEÇİMİ

Bu çalışmada, Türkiye'de Fiat marka Egea model otomobillerin satış fiyatlarına etki eden faktörlerin incelenmesi amacıyla www.sahibinden.com sitesinin veri tabanından elde edilen veriler kullanılmıştır. Çalışmanın örneklemini, 20 Ocak 2024 ile 20 Şubat 2024 tarihleri arasında ilanda bulunan 702 adet Fiat Egea otomobil oluşturmaktadır. Bu dönemde, Fiat Egea modeline ait olan satış ilanları detaylı bir şekilde incelenmiş, ilanlarda yer alan araçların yaş, kilometre, yakıt

türü, vites tipi, kasa tipi gibi temel özellikleri başta olmak üzere ekstra özellikler de dikkate alınarak kapsamlı bir veri seti oluşturulmuştur.

Analiz sonuçlarının karşılaştırılması ve değerlendirilmesi aşamasında kolaylık sağlaması amacıyla örneklem, Türkiye'de en yüksek satış oranına sahip olan Fiat Egea modeli ile sınırlandırılmıştır. Analizde kullanılan değişkenlere ait tanımlar ve kategoriler Tablo 6.1'de detaylı olarak verilmektedir. Bu tablo, çalışmanın yöntemi kısmında açıklanan Hedonik Fiyat Modeli'ne uygun olarak düzenlenmiştir ve SPSS programı kullanılarak değerlendirilmiştir. Modelin doğruluğu, çeşitli istatistiksel testlerle sınanmış ve onaylanmıştır. Bu sayede, Fiat Egea model otomobillerin satış fiyatları üzerinde etkili olan bağımsız değişkenlerin yüzdelik etkileri belirlenmiş ve ilgili literatürle karşılaştırmalar yapılmıştır.

Bu kapsamlı veri toplama ve analiz süreci, Türkiye otomotiv piyasasındaki fiyat dinamiklerini daha iyi anlamak ve potansiyel alıcılar ile satıcılar için yararlı bilgiler sunmak amacıyla tasarlanmıştır.

Tablo 1. YSA Dog-Dog Sonuçları ile Hedonik Dog-Dog Sonuçlarının Performans Karşılaştırma Ölçütlerine Göre Karşılaştırılmasına Ait Sonuçlar

Yöntem	Gizli Katman 1		Gizli Katman 2		Gizli Katman 3		RMSE	MAPE	MAD
	Nöron	Akt.	Nöron	Akt.	Nöron	Akt.			
YSA (Dog-Dog)	**1024**	**tanh**	**512**	**tanh**	**512**	**tanh**	**96852,7**	**0,11**	**71425,66**
	128	tanh	64	tanh	Yok	Yok	116665,25	0,22	106692,77
	64	tanh	32	tanh	Yok	Yok	230475,4	0,22	178064,52
	32	tanh	16	tanh	Yok	Yok	244476,85	0,24	231071,12
	16	tanh	8	tanh	Yok	Yok	244258,13	0,28	225526,4
	8	tanh	4	tanh	Yok	Yok	262008,6	0,32	288024,4
	4	tanh	2	tanh	Yok	Yok	269057,8	0,33	288078,7
YSA (Dog-Dog)	1024	relu	512	relu	512	relu	102652,4	0,12	75232,95
	128	relu	64	relu	Yok	Yok	135715,2	0,16	277920,8
	64	relu	32	relu	Yok	Yok	310475,5	0,16	164664,14
	32	relu	16	relu	Yok	Yok	204476,12	0,19	191202,3
	16	relu	8	relu	Yok	Yok	242899,6	0,22	214085,89
	8	relu	4	relu	Yok	Yok	280522,2	0,22	295072,1
	4	relu	2	relu	Yok	Yok	156666,12	0,28	159017,14
Hedonik (Dog-Dog)	Yok	Yok	Yok	Yok	Yok	Yok	211612,6	0,13	88322,16

Tablo 2. YSA Log-Dog Sonuçları ile Hedonik Log-Dog Sonuçlarının Performans Karşılaştırma Ölçütlerine Göre Karşılaştırılmasına Ait Sonuçlar

Yöntem	Gizli Katman 1		Gizli Katman 2		Gizli Katman 3		RMSE	MAPE	MAD
	Nöron	Akt.	Nöron	Akt.	Nöron	Akt.			
YSA(Log-Dog)	**1024**	**tanh**	**512**	**tanh**	**512**	**tanh**	**76888,1**	**0,10**	**65456,02**
	128	tanh	64	tanh	Yok	Yok	232615,2	0,24	207921,88
	64	tanh	32	tanh	Yok	Yok	193870,14	0,21	184289,22
	32	tanh	16	tanh	Yok	Yok	24476,85	0,23	231071,12
	16	tanh	8	tanh	Yok	Yok	244258,13	0,27	225526,4
	8	tanh	4	tanh	Yok	Yok	262008,6	0,31	288024,4
	4	tanh	2	tanh	Yok	Yok	269057,8	0,32	288078,7
YSA (Log-Dog)	1024	relu	512	relu	512	relu	114697,54	0,12	87170,22
	128	relu	64	relu	Yok	Yok	345715,6	0,15	337920,8
	64	relu	32	relu	Yok	Yok	331414,14	0,15	324464,14
	32	relu	16	relu	Yok	Yok	245476,55	0,22	238071,12
	16	relu	8	relu	Yok	Yok	265258,44	0,26	205526,6
	8	relu	4	relu	Yok	Yok	212008,6	0,26	198024,4
	4	relu	2	relu	Yok	Yok	256158,6	0,28	242455,06
Hedonik (Log-Dog)	Yok	Yok	Yok	Yok	Yok	Yok	115442,6	0,11	77564,14

Tablo 3. Performans Ölçüm Kriterine Göre En İyi Sonuçları Veren Modelin Belirlenmesine Ait Dağılım

Model	Gizli Katman 1		Gizli Katman 2		Gizli Katman 3		RMSE	MAPE	MAD
	Nöron	Akt.	Nöron	Akt.	Nöron	Akt.			
YSA (Dog-Dog)	1024	tanh	512	tanh	512	tanh	96852,7	0,11	71425,66
YSA (Log-Dog)	**1024**	**tanh**	**512**	**tanh**	**512**	**tanh**	**76888,1**	**0,10**	**65456,02**
YSA (Dog-Dog)	1024	relu	512	tanh	512	relu	102652,4	0,12	75232,95
YSA (Log-Dog)	1024	relu	512	tanh	512	relu	114697,54	0,12	87170,22
Hedonik(Dog-Dog)	Yok	Yok	Yok	Yok	Yok	Yok	211612,6	0,13	88322,16
Hedonik(Log-Dog)	Yok	Yok	Yok	Yok	Yok	Yok	115442,6	0,11	77564,14

Tablo 1'de YSA Dog-Dog sonuçları ile Hedonik Dog-Dog sonuçlarının performans karşılaştırma ölçütlerine göre karşılaştırılması verilmiştir. YSA kul lanılan bu çalışmada 1 girdi katmanı, 3 gizli katman ve 1 çıktı katmanı kul lanılmıştır. Analiz aşamasında tanh ve relu olmak üzere iki adet aktivason fonksiyonu kullanılmıştır. Katmanlardaki nöron sayıları 2, 4, 8, …, 1024 şeklinde arttırılarak sonuçlar elde edilmiştir. YSA

(dog-dog) modelinde, birinci gizli katman sayısının 1024, ikinci gizli katman sayısının 512, üçüncü gizli katman sayısının 512 olduğu durumda ve aktivasyon fonksiyonu olarak tanh kull lanıldığında performans değerlendirme kriterleri şu şekilde elde edilmiştir: RMSE = 96852,7, MAPE = 0,11 ve MAD = 71425,66.

Tablo 2'de YSA Log-Dog sonuçları ile Hedonik Log-Dog sonuçlarının performans karşılaştırma ölçütlerine göre karşılaştırılması verilmiştir. YSA (Log-Dog) modelinde, birinci gizli katman sayısının 1024, ikinci gizli katman sayısının 512, üçüncü gizli katman sayısının 512 olduğu durumda ve aktivasyon fonksiyonu olarak tanh kullanıldığında performans değerlendirme kriterleri şu şekilde elde edilmiştir: RMSE = 76888,1, MAPE = 0,10 ve MAD = 65456,02. Hedonik (Log-Dog) modelinde, RMSE = 115442,6, MAPE = 0,11 ve MAD = 77564,14 olarak elde edilmiştir. Bu tabloda performans kriterleri değerlendirildiğinde en iyi sonucu, YSA (Log-Dog) modelinde tanh aktivasyon fonksiyonu kullanıldığında RMSE değerinin verdiği görülmektedir.

Tablo 3'te, en iyi performans kriteri değerlerini veren model sonuçları verilmiştir. Analiz bulgularına göre, en iyi performans kriter değerini YSA (Log-Dog) modelinin verdiği görülmektedir.

Bu sonuçlar, çalışmanın amacına uygun olarak hedonik fiyat modeli ve yapay sinir ağları yöntemlerinin karşılaştırmalı analizini sunmakta ve hangi yöntemin daha iyi performans gösterdiğini belirlemektedir.

5. SONUÇ VE ÖNERİLER

Bu çalışmada, Türkiye'de en çok satılan otomobilin fiyatına etki eden faktörler, Hedonik Fiyat Modeli ve Yapay

Sinir Ağları (YSA) kullanılarak analiz edilmiştir. İkinci el otomobil fiyatlandırması, hem satıcılar hem de alıcılar için zorlu bir süreçtir. Bu bağlamda, çalışmanın temel amacı Türkiye'de en çok satılan otomobillerin fiyatlarını belirlemede etkili olan faktörleri istatistiksel yöntemlerle ortaya koymaktır. Bunun yanı sıra, Hedonik Fiyat Modeli ve YSA yöntemlerinin otomobil fiyatı belirlemedeki tahmin performanslarını karşılaştırmaktır.

Çalışma kapsamında 702 adet otomobil analize dahil edilmiştir. Otomobillerin özellikleri ve yapılan literatür taraması derlenerek, otomobil fiyatlarına etki eden bağımsız değişkenler belirlenmiş ve veri seti oluşturulmuştur. Oluşturulan veri seti ile SPSS programı üzerinden Hedonik Fiyat Modeli ile analizler yapılmıştır. Modelin doğruluğu ilgili testler ile sınanmış ve onaylanmıştır. Yapılan analiz ile bağımlı değişken olan fiyata yüzdelik etkiler belirlenen bağımsız değişkenler ile hesaplanmıştır.

Çalışmanın bulgularına göre, otomobil fiyatını etkileyen önemli faktörler arasında marka, otomobil yaşı, yakıt türü, vites türü, kilometre, kasa tipi, motor gücü, motor hacmi, garanti, takas, sahibinden satış, tramer kaydı, boyalı ve değişen parçalar, start-stop özelliği, geri görüş kamerası, dijital klima, anahtarsız çalıştırma, ve sis farı gibi özellikler yer almaktadır. Bu faktörlerin her biri, otomobilin fiyatında belirli oranlarda artış veya azalışa sebep olmaktadır.

Hedonik Fiyat Modeli ve YSA sonuçları karşılaştırıldığında, YSA'nın daha yüksek bir tahmin doğruluğuna sahip olduğu görülmüştür. YSA (Log-Dog) modelinde, birinci gizli katman sayısının 1024, ikinci gizli katman sayısının 512, üçüncü gizli katman sayısının 512 olduğu ve aktivasyon fonksiyonu olarak tanh kullanıldığında, RMSE (Kök Ortalama Kare Hatası) değeri 76888,1, MAPE (Ortalama Mutlak Yüzdesel Hata) değeri 0,10 ve MAD (Medyan

Çevresinde Mutlak Sapma) değeri 65456,02 olarak elde edilmiştir. Hedonik (Log-Dog) modelinde ise bu değerler sırasıyla 115442,6, 0,11 ve 77564,14 olarak bulunmuştur. Bu sonuçlar, YSA'nın hedonik modellere kıyasla daha iyi bir performans sergilediğini göstermektedir.

Bu çalışma, Türkiye'de en çok satılan otomobil fiyatlarının belirlenmesinde YSA yönteminin hedonik modellere göre daha etkili bir yöntem olabileceğini ortaya koymuştur. İlerleyen çalışmalarda, farklı otomobil segmentleri ve farklı bölgesel pazarlar için benzer analizler yapılabilir. Ayrıca, diğer makine öğrenme algoritmaları ve modelleri de bu tür analizlerde kullanılabilir.

Sonuç olarak, ikinci el otomobil fiyatlarının belirlenmesinde kullanılan yöntemlerin performanslarının karşılaştırılması, hem akademik literatüre katkı sağlamış hem de pratik uygulamalarda daha doğru fiyat tahminleri yapılmasına olanak tanımıştır. Bu çalışma, otomobil pazarında faaliyet gösteren işletmelere, satış stratejilerini belirlerken ve fiyatlandırma yaparken önemli bilgiler sunmaktadır.

KAYNAKLAR

Akbaba, H.K. (2019). Beyaz Akaryakıt Satış Miktarının Yapay Sinir Ağı ile Modellenmesi. Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstatistik Anabilim Dalı, İstatistik Programı, Samsun.

Aliefendioğlu, Y. (2011). Türkiye'de Koruma Alanlarındaki Taşınmazların Kullanımı ve Koruma Statülerinin Taşınmaz Piyasaları ve Değerlerine Etkileri: Muğla İli Örneği. (Yayımlanmış Doktora Tezi), Ankara: Ankara Üniversitesi Fen bilimleri enstitüsü

Alkay, E., (2002). "Hedonik Fiyat Yöntemi İle Kentsel Yeşil Alanların Ekonomik Değerlerinin Ölçülmesi", Yayınlanmamış Doktora Tezi. İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü.

Arslan, K. (2003). Otomobil alımında tüketici davranışlarını etkileyen faktörler. İstanbul Ticaret Üniversitesi Dergisi, 3, 83-103.

Arıkan, F. E. (2008). Ev Kiralarını Etkileyen Faktörlerin Hedonik Fiyat Yöntemi İle Belirlenmesi. (Yayımlanmış Yüksek Lisans Tezi), İstanbul: Marmara Üniversitesi Sosyal Bilimler Enstitüsü

Ataseven, B. (2013). Yapay sinir ağları ile öngörü modellemesi. *İstanbul Kültür Üniversitesi, İktisadi ve İdari Bilimler Fakültesi Dergisi,* 10(39), 101-115.

Ateş, M. Ş. (2023). Samsun'da ikinci el otomobil fiyatlarını etkileyen faktörlerin tahmininde hedonik fiyat ve yapay sinir ağları modellerinin karşılaştırılması. (Yayımlanmış Yüksek Lisans Tezi), Ondokuz Mayıs Üniversitesi Lisansüstü Eğitim Enstitüsü İstatistik Ana Bilim Dalı, Samsun.

Bıçakcı, P. S., ve Üreten S. (2017). Tedarik zinciri yönetimi uygulamalarının zincir performansı üzerindeki etkileri: Bir uygulama. *Gazi Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 19(1), 367-386.

Cingöz, A., (2011). Hedonik Talep Teorisi Çerçevesinde Bir Fiyatlandırma Örneği, Yayımlanmamış Doktora Tezi, İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü, İktisat Bölümü, İstanbul.

Düzgün, K. (2022). Konut Fiyatlarına Etki Eden Faktörlerin Hedonik Modelle Belirlenmesi, Ondokuz Mayıs Üniversitesi Lisansüstü Eğitim Enstitüsü Harita Mühendisliği Ana Bilim Dalı, Samsun.

Ecer, F. (2013). Türkiye'de 2. el otomobil fiyatlarının tahmini ve fiyat belirleyicilerinin tespiti. Anadolu Üniversitesi Sosyal Bilimler Enstitüsü Dergisi, (11), 101-12.

Emre Aysin, M. (2018). Konut Fiyatlarına Etki Eden Faktörlerin Hedonik Modelle Belirlenmesi:TRA1 Alt Bölgesi Üzerine Bir Uygulama. Erzurum Teknik Üniversitesi Sosyal Bilimler Enstitüsü, İktisat Anabilim Dalı, Erzurum.

Gencel, U. ve Kuru, E. (2012). Vergi kültürü ve vergi politikaları etkileşimi: Türkiye değerlendirmesi. *Yönetim Bilimleri Dergisi*, 10(20), 29-60.

Güneş, S. (2012). Türk Toplumu ve Otomobil. SDÜ Fen Edebiyat Fakültesi Sosyal Bilimler Dergisi, 1(25), 213-230.

Hepşen, A. (2011). Finansal Krizlerde Gayrimenkul Finansman Piyasalarının Rolü ve Gayrimenkul Fiyat Endekslerinin Önemi. İstanbul: Literatür Yayıncılık

Kaya, T. (2023). Mahalle Ölçeğinde Konut Fiyatlarına Etki Eden Faktörlerin Hedonik Fiyat Modeli ile Analiz

Edilmesi: Esenkent Mahallesi Ölçeği. Mimar Sinan Güzel Sanatlar Üniversitesi, Fen Bilimleri Enstitüsü Yapı Mühendisliği Anabilim Dalı Yapım Proje Yönetimi Programı, İstanbul.

Akbaba, H. (2019). Beyaz akaryakıt satış miktarının yapay sinir ağları ile modellenmesi. (Yayımlanmış Yüksek Lisans Tezi), Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü İstatistik Anabilim Dalı İstatistik Programı, İstanbul.

Kördiş, G., Işık S., ve Mert M., (2014). Antalya'da Konut Fiyatlarını Etkileyen Faktörlerin Hedonik Fiyat Modeli İle Tahmin Edilmesi, Akdeniz İ.İ.B.F. Dergisi.

Krenker, A., Bešter, J., ve Kos, A. 2011. Introduction to the Artificial Neural Networks, In: Artificial Neural Networks-Methodological Advances and Biomedical Applications. Suzuki, K. (editör). Intech, 1-18, Londra.

Oruç, E. N. (2021). Yapay Sinir Ağları Kullanılarak Kısa Süreli Güneş Enerjisi Tahmini. İstanbul Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü, Meteoroloji Mühendisliği Anabilim Dalı Atmosfer Bilimleri Programı, İstanbul.

Özçalıcı, M. (2017). Özdüzenleyici Haritalar Yardımıyla Piyasa Bölümlendirmesi: Türkiye İkinci El Otomobil Piyasası Örneği. *Eskişehir Osmangazi Üniversitesi İktisadi ve İdari Bilimler Dergisi*, 12(2), 23-36.

Özgür, B. (2023). Türkiye'de Faaliyet Gösteren Bazı Sigorta Şirketlerinin Prim Üretimlerinin Yapay Sinir Ağları İle Tahmini, Marmara Üniversitesi Fen Bilimleri Enstitüsü, İstatistik Anabilim Dalı İstatistik Programı, İstanbul.

Sarker, I. H. (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and

Research Directions. SN Computer Science 2:420 https://doi.org/10.1007/s42979-021-00815-1.

Satoğlu, G. (2023). Metro İstasyonlarının Gayrimenkul Değerlerine Etkisinin Hedonik Fiyat Modeli ile Değerlendirilmesi: Batıkent ve Koru Örnekleri. Gazi Üniversitesi Fen Bilimleri Enstitüsü Şehir ve Bölge Planlama Anabilim Dalı, Ankara.

Taştan, N. ve Taştan, K. (2023). Otomotiv sektörü 2017-2023: Küresel bir analiz ve Türkiye. *Stratejik Yönetim Araştırmaları Dergisi*, 6(2), 120-143.

Tutar, Ü., ve Çamlıbel F. (2022). Tüketici Tercihini Etkileyen Tüketici Değer Algısının Kavramsal Boyutları. *Balıkesir Üniversitesi* İktisadi *ve İdari Bilimler Fakültesi Dergisi,* , 3(1), 47-58.

Uluscul, C., ve Demir, A. O. (2023). Otomobil tercihinde tüketici eğilimleri, Türkiye'de yeni ve ikinci el otomobil satınalma kararı. *İstanbul Ticaret Üniversitesi Girişimcilik Dergisi.* 7(13), 64-86.

Ulusoy, A., ve Ela, M. (2016). Sukukun ikinci el piyasa sorunu. *Hukuk Ve İktisat Araştırmaları Dergisi*, 8(1), 55-70.

Yayar, R. (2011). Dizüstü Bilgisayar Piyasasında Hedonik Talep Parametrelerinin Tahminlenmesi, KMÜ Sosyal ve Ekonomik Araştırmalar Dergisi, ISSN: 1309– 913213, 13(21): 21–27.

E-TİCARET SİTELERİNDE ÜRÜN TAVSİYE SİSTEMLERİ

Melih ÜSTÜN[1]

Erol TERZİ[2]

Mehmet Şirin ATEŞ[3]

1. GİRİŞ

Tavsiye sistemleri, modern dijital çağda müşteri deneyimini kişiselleştirme ve işletmelerin karını artırmak için önemli bir rol oynamaktadır. Bu sistemler, kullanıcıların ilgi alanlarını, tercihlerini ve davranışlarını analiz ederek onlara özel ürün, hizmet veya içerik önerileri sunar. E-ticaret siteleri, müşteri memnuniyetini artırmak, satın almaya yönlendirmek ve kullanıcı bağlılığını güçlendirmek için tavsiye sistemlerini yaygın olarak kullanmaktadır.

Tavsiye sistemlerinin temel amacı, müşterilerin taleplerini daha iyi anlamak ve onlara en uygun alternatifleri sunmak ve karar verme süreçlerini kolaylaştırmaktır. Bu sayede kullanıcılar, aradıkları ürünleri kolayca bulabilir, yeni ilgi alanları keşfedebilir ve platformda daha fazla zaman geçirebilirler. Firmalar açısından ise tavsiye sistemleri, müşteri

[1] Lisansüstü Öğrenci, Ondokuz Mayıs Üniversitesi, Fen Fakültesi, İstatistik Bölümü, 22283100@stu.omu.edu.tr, ORCID: 0009-0009-6104-5206.

[2] Prof. Dr., Ondokuz Mayıs Üniversitesi, Fen Fakültesi, İstatistik Bölümü, eroltrz@omu.edu.tr, ORCID: 0000-0002-2309-827X.

[3] Arş.Gör., Ondokuz Mayıs Üniversitesi, Fen Fakültesi, İstatistik Bölümü, mehmet.ates@omu.edu.tr, ORCID: 0000-0001-9904-6380.

memnuniyetini artırarak sadık bir müşteri kitlesi oluşturmanın yanı sıra satışları ve gelirleri artırmak için etkili bir araçtır.

Bu çalışmada, e-ticaret sitelerinde kullanılan ürün tavsiye sistemleri detaylı bir şekilde incelenecektir. Tavsiye sistemlerinin temeli, farklı türleri, çalışma prensipleri, avantajları ve dezavantajları islenecektir. Ayrıca, tavsiye sistemlerinin karşılaştığı zorluklar ve bu zorlukların üstesinden gelmek için geliştirilen çözüm yolları da tartışılacaktır. Özellikle, soğuk başlangıç problemi, veri seyrekliği, ölçeklenebilirlik, filtre balonu ve gizlilik gibi konular üzerinde durulacaktır.

Bu çalışma, e-ticaret sitelerinde ürün tavsiye sistemlerinin çeşitliliğini ortaya koyarak, bu alandaki araştırmalara ve uygulamalara katkı sağlamayı amaçlamaktadır. İşletmelerin, kendi özel ihtiyaçlarına ve hedeflerine uygun olan en iyi tavsiye sistemi yöntemini seçebilmeleri için farklı yöntemlerin avantajları ve dezavantajları karşılaştırılacaktır.

2. TAVSİYE SİSTEMLERİ

2.1. Tavsiye Sistemlerinin Kullanım Amaçları

Günümüz dijital çağında, müşteriler bilgi ve ürün bolluğunda kaybolmadan ihtiyaçlarına uygun olan ürünü bulmaları giderek zorlaşıyor. İşte bu noktada, tavsiye sistemleri devreye girerek kişiselleştirilmiş deneyimler sunmayı amaçlıyor. Hem kullanıcıların hem de işletmelerin hayatını kolaylaştırıyor.

2.1.1. Kişiselleştirilmiş Deneyim Sunma

Her kullanıcı benzersizdir ve değişik beklentileri vardır. Tavsiye sistemleri, kullanıcıların geçmiş davranışlarını, tercihlerini ve ilgi alanlarını analiz ederek onlara has tavsiyeler sunar. Bu sayede kullanıcılar, kendileri için en uygun olanları

daha hızlı ve kolay bir şekilde bulabilme imkanına ulaşırlar . Günümüzde çoğu web sitesi bu yöntemleri kullanarak müşteriye has tavsiyeler sunma kabiliyetine sahipler.

2.1.2. Kullanıcı Etkileşimini Artırma

Tavsiye sistemleri, kullanıcıların platforma olan ilgisini canlı tutar ve etkileşimde kalmasını sağlar. Nokta atışı tavsiyeler, kullanıcıların sitede daha fazla zaman geçirmesini ve daha fazla içerik tüketmesini neden olur. Örneğin; Netflix'in kişiselleştirilmiş film ve dizi tavsiyeleri, kullanıcıların platformda daha fazla vakit geçirmesine ve daha çok içerik tüketmesine yardımcı olur.

2.1.3. Satışları Artırma

E-ticaret sitelerinde kullanılan tavsiye sistemleri için aslında temel amaç olan satışları artırmak ve karlılığı maksimize etmektir. Bu sistemler kullanıcılara kişiselleştirilmiş ürün tavsiyeleri sunarak satın alma olasılıklarını yükseltir. Örneğin, "Sizin İçin Tavsiyelenler" veya "Bunları da Beğenebilirsiniz" gibi bölümler, kullanıcıların daha fazla ürün keşfetmesini ve satın almasını teşvik eder. (Schafer et al., 1999)

2.1.4. Müşteri Sadakatini Güçlendirme

Kişiselleştirilmiş tavsiyeler, kullanıcıların kendilerini özel ve değerli hissetmelerini sağlar. Kullanıcılar, platformun kendilerini anladığını ve önemsediğini düşündüklerinden, web sitesine olan bağlılıkları artar ve tekrar ziyaret etme olasılıkları yükselir. Spotify'ın kişiselleştirilmiş çalma listeleri, kullanıcıların platforma olan bağlılıklarını artırarak sadık bir kullanıcı kitlesi oluşturmasına yardımcı oluyor. (Schedl et al., 2018)

2.1.5. Keşif İmkanı Sunma

Tavsiye sistemlerinin bir diğer, kullanım amacı müşterilerin sayfada kolayca bulamayacakları yeni ürünler,

içerikler veya hizmetlere ulaşmalarını sağlamaktır. Bu, kullanıcıların ilgi alanlarını genişletmelerine ve platformun sunduğu çeşitliliği keşfetmelerine olanak tanır.

Örneğin, YouTube'un video tavsiyeleri, kullanıcıların yeni kanallar ve içerikler keşfetmesine yardımcı olur .

2.1.6. Verimlilik ve Karlılık Artışı

Tavsiye sistemleri, aynı zamanda pazarlama ve satış süreçlerde kullanılarak süreç optimizasyonu ile verimlilik artışına neden olur . Doğru hedef kitleye doğru zamanda doğru tavsiyeler sunarak, kampanya maliyetlerini düşürür . Örneğin, bir giyim markası kişiselleştirilmiş e-posta kampanyaları ile satışa dönüşen kampanya oranını arttırabilir.

2.2. Tavsiye Sistemlerinin Kullanım Amaçları

Tavsiye sistemleri, modern dijital dünyanın kullanıcı deneyimini kişiselleştirmenin ve işletmelerin başarısını artırmanın vazgeçilmez bir parçasıdır. Bu sistemler, kullanıcıların ihtiyaçlarını daha iyi anlamak ve onlara en uygun çözümleri sunmak için sürekli olarak geliştirimektedir.

Tavsiye sistemleri, yapay zeka ve makine öğrenmesi alanındaki gelişmelerle birlikte daha da sofistike hale geliyor. Gelecekte, daha doğru ve kişiselleştirilmiş tavsiyeler sunan, kullanıcıların duygusal tepkilerini dikkate alan ve gerçek zamanlı olarak etkileşimde bulunan tavsiye sistemleri görmek hiç şaşırtıcı olmayacaktır .

2.2.1. Kullanıcılar Açısından Tavsiye Sistemlerinin Faydaları

2.2.1.1.Zaman tasarrufu sağlar ve daha kolay karar vermeye yardımcı olurlar.

2.2.1.2.İlgi alanlarına uygun içeriklere erişimi kolaylaştırırlar.

2.2.1.3. Yeni keşifler yapma fırsatı verirler

2.2.1.4. Platformla etkileşimi arttırarak memnuniyeti seviyeleri arttırdılar.

2.2.2. İşletmeler Açısından Tavsiye Sistemlerinin Faydaları

2.2.2.1. Satışların artması ve müşteri sadakatinin güçlenmesi konusunda etkilidirler.

2.2.2.2. Pazarlama ve reklam maliyetlerinin düşürürler

2.2.2.3. Müşteri davranışları konusunda detaylı bilgi verirler.

2.2.2.4. Rekabet avantajı sağlarlar

3. TAVSİYE SİSTEMLERİNDE KULLANILAN YÖNTEMLER

3.1. Basit Tavsiye Sistemleri

E-ticaret platformlarında kişiye özel tavsiyeler, müşteri memnuniyetini ve satışları artırmak için büyük öneme sahiptir. Basit tavsiye sistemleri, bu kişiselleştirme yolculuğunda önemli bir başlangıç noktası sunar. Karmaşık algoritmalar veya derinlemesine analiz gerektirmeyen bu sistemler, temel prensiplerle çalışarak kullanıcıların karşısına ürünleri çıkarır. Özellikle yeni başlayan veya daha küçük ölçekli e-ticaret platformları için ideal bir seçenektirler.

Basit tavsiye sistemleri, e-ticaret siteleri için her derde deva bir çözüm olmasa da, kişiselleştirme yolculuğunda önemli bir adımdır. Sınırlı kişiselleştirme yeteneklerine rağmen, kullanıcı deneyimini iyileştirmek ve satışları artırmak için etkili bir şekilde kullanılabilirler. Özellikle yeni başlayan veya daha küçük ölçekli e-ticaret platformları için başlangıç noktası olarak

değerlendirilebilirler. Ancak, rekabette öne çıkmak ve kullanıcı memnuniyetini en üst düzeye çıkarmak için, zamanla daha gelişmiş ve kişiselleştirilmiş tavsiye sistemlerine geçiş yapılması faydalı olacaktır.

3.1.1. Çalışma Prensipleri

Basit tavsiye sistemleri, aşağıdaki şekilde temel tavsiyelerden oluşur;

- **Popülerlik Tabanlı Tavsiyeler:** En çok satan, en çok görüntülenen veya en yüksek puan alan ürünleri kullanıcılara tavsiye eder. Bu yöntem, genel eğilimlere ve popüler tercihlere odaklanır.

- **Rastgele Tavsiyeler:** Ürünleri rastgele seçerek kullanıcılara sunar. Bu yöntem, keşif ve sürpriz unsuru yaratmayı amaçlar. (Örnek: Bazı e-ticaret sitelerindeki "Sizin İçin Seçtiklerimiz" bölümleri)

- **Demografik Tabanlı Tavsiyeler:** Kullanıcıların yaş, cinsiyet, konum, ilgi alanları gibi demografik bilgilerine dayanarak ürün tavsiyeleri yapar. Bu yöntem, kullanıcı segmentasyonuna odaklanır. (Örnek: Netflix'in demografik bilgilere göre içerik tavsiyeleri) (Zhang et al., 2015)

- **Yeni Ürün Tavsiyeleri:** En yeni eklenen ürünleri kullanıcılara tanıtır. Bu yöntem, yenilik ve çeşitlilik sunmayı amaçlar. Aynı zamanda reklam maliyetini düşürür.

- **İndirimli Ürün Tavsiyeleri:** İndirimli veya kampanyalı ürünleri öne çıkarılır. Bu yöntem, kullanıcıların fiyat odağını hedefler.

3.1.2. Avantajları

- **Uygulama Kolaylığı:** Hızlı ve kolay bir şekilde uygulanabilirler.
- **Düşük Maliyet:** Geliştirme ve bakım maliyetleri düşüktür.
- **Hızlı Sonuç:** Anında ürün tavsiyeleri sunabilirler.

3.1.3. Dezavantajları

- **Sınırlı Kişiselleştirme:** Kullanıcıların bireysel tercihlerini ve ilgi alanlarını derinlemesine analiz yetenekleri yoktur.
- **Öğrenme Yeteneği Olmaması:** Kullanıcı davranışlarından öğrenerek tavsiyeleri iyileştirme yetenekleri sınırlıdır
- **Rekabet Dezavantajı:** Daha gelişmiş kişiselleştirme teknikleri kullanan rakiplere göre dezavantajlı olabilirler.

3.1.4. Kullanım Alanları

- **Yeni Başlayan E-Ticaret Siteleri:** Kişiselleştirmeye başlamak, ilk adımı atmak isteyenler için uygun bir başlangıç noktasıdır.
- **Küçük Ölçekli E-Ticaret Platformları:** Sınırlı kaynaklara sahip siteler için uygun bir seçenektir.
- **Büyük E-Ticaret Sitelerinin Belirli Bölümleri:** Ana sayfada popüler ürünleri öne çıkarmak veya yeni ürünleri tanıtmak gibi belirli amaçlar için kullanılabilirler.

3.2. Birliktelik Kuralı Öğrenimi

Birliktelik kuralı öğrenimi (Association Rule Learning - ARL), e-ticaret sitelerinde ürünler arasındaki ilişkileri ortaya çıkarmak için kullanılan güçlü bir veri madenciliği tekniğidir. Bu yöntem, müşterilerin hangi ürünleri birlikte satın alma veya

görüntüleme eğiliminde olduğunu analiz ederek, pazarlama ve satış stratejilerini geliştirmek için değerli bilgiler sunar (Tan, Steinbach, & Kumar, 2005).

Bir e-ticaret sitesinde bebek bezi alan müşterilerin %60'ının aynı zamanda ıslak mendil aldığı tespit edilirse, bu bir birliktelik kuralıdır. Bu kuralın destek değeri yüksekse (örneğin, %20), bebek bezi ve ıslak mendilin sıklıkla birlikte satın alındığını gösterir. Güven değeri de yüksekse (örneğin, %60), bebek bezi alan müşterilerin ıslak mendil alma olasılığının yüksek olduğunu gösteir.

Bu bilgi, e- ticaret sitesinin bebek bezi ve ıslak mendili birlikte önermesine, bu ürünleri yan yana sergilemesine veya bu ürünler için özel bir kampanya oluşturmasına yardımcı olabilir.

Birliktelik kuralı öğrenimi, e -ticaret sitelerinin müşteri davranışlarını daha iyi anlamalarına ve bu bilgileri pazarlama ve satış stratejilerini geliştirmek için kullanmalarına olanak tanıyan güçlü yontemdır. Bu yöntem, kişiselleştirilmiş ürün tavsiyeleri sunarak, çapraz satış ve yukarı satış fırsatlarını belirleyerek, mağaza düzenini optimize ederek ve stok yönetimini iyileştirerek e- ticaret sitelerinin gelirlerini artırmalarına yardımcı olabilir.

3.2.1. Temel Kavramlar

- **Öğe (Item):** E-ticaret bağlamında bir ürün veya hizmettir

- **İşlem (Transaction):** Bir müşterinin belirli bir zamanda yaptığı satın alma veya görüntüleme eylemidir

- **Birliktelik Kuralı (Association Rule):** İki veya daha fazla öğe arasındaki ilişkiyi ifade eder. Örneğin, "X ürününü alan müşterilerin %70'i Y ürününü de alıyor" gibi bir kuraldır (Hahsler, Grün, & Hornik, 2005).

- **Destek (Support):** Bir kuraldaki öğelerin birlikte kaç işlemde geçtiğini gösterir. Yüksek destek, öğelerin sıklıkla birlikte satın alındığını veya görüntülendiğini gösterir (Hahsler et al., 2005).

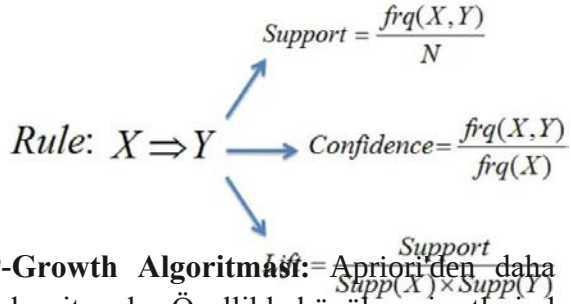
- **Güven (Confidence):** Bir kuralın ne kadar güvenilir olduğunu gösterir. Örneğin, %70 güven, X ürününü alan müşterilerin % 70'inin Y ürününü de aldığını gösterir (Hahsler et al., 2005).

- **Lift (Kaldırma):** Bir kuralın rastgele birlikteliklerden ne kadar farklı olduğunu gösterir. Lift değeri 1'den büyükse, kuralın rastgele birliktelikten daha anlamlı olduğunu gösterir (Hahsler et al., 2005)

3.2.2. Algoritmalar

Birliktelik kuralı öğrenimi için kullanılan en yaygın algoritmalar şunlardır:

- **Apriori Algoritması:** En temel ve sık kullanılan algoritmadır. Belirli bir destek ve güven eşiğinin üzerindeki tüm birliktelik kurallarını bulur. En çok kullanılan algoritmadır. (Agrawal & Srikant, 1994).



- **FP-Growth Algoritması:** Apriori'den daha hızlı ve verimli bir algoritmadır. Özellikle büyük veri setlerinde daha iyi performans gösterir (Han, Pei, & Yin, 2000).

- **Eclat Algoritması:** Apriori'ye benzer ancak farklı bir veri yapısı kullanır. Bazı durumlarda Apriori'den daha hızlı olabilir (Zaki, 2000).

3.2.3. Kullanım Alanları

- **Sepet Analizi:** Müşterilerin sepetlerindeki ürünleri analiz ederek çapraz satış ve yukarı satış fırsatlarını belirler (Şengör, 2019).
- **Mağaza Düzeni Optimizasyonu:** Birlikte satın alınan ürünleri birbirine yakın yerleştirerek müşteri deneyimini iyileştirir (Kültür & Özmen, 2016).
- **Kampanya Tasarımı:** Birlikte satın alınan ürünler için özel kampanyalar oluşturarak satışları artırır (Kültür & Özmen, 2016).
- **Stok Yönetimi:** Birlikte satın alınan ürünlerin stok seviyelerini analiz ederek stok kesintilerini onler (Kültür & Özmen, 2016).

3.3. İçerik Tabanlı Filtreleme

İçerik tabanlı filtreleme (Content-Based Filtering - CBF), e-ticaret sitelerinde müşterilerin bireysel tercihlerine ve ilgi alanlarına göre kişiselleştirilmiş ürün tavsiyeleri sunmak için kullanılan etkili bir yöntemdir (Pazzani & Billsus, 2007). Bu yaklaşım, ürünlerin özelliklerini ve kullanıcıların geçmişte beğendiği veya etkileşimde bulunduğu ürünlerin özelliklerini analiz ederek benzer ürünleri tavsiye eder (Lops, de Gemmis, & Semeraro, 2011).

Örnek olarak, bir müşteri daha önce telefon ve kulaklık satın aldıysa, içerik tabanlı filtreleme bu kullanıcının profilini "elektronik" olarak işaretleyebilir ve benzer ürünleri önerebilir.

İçerik tabanlı filtreleme, e-ticarette kullanıcı memnuniyetini artırmak ve satışları yükseltmek için güçlü bir araçtır . Ürün tavsiyeleri sunarak, müşterilerin ilgisini çeken ve ihtiyaçlarını karşılayan ürünleri bulmalarına yardımcı olur. Ancak, tek başına kullanıldığında sınırlı keşif imkanı sunabilir Bu nedenle, diğer tavsiye sistemleri yöntemleriyle birlikte kullanılması, örneğin işbirlikçi filtreleme ile hibrit bir model oluşturulması, daha etkili sonuçlar verebilir (Claypool et al., 1999).

3.3.1. Çalışma Prensibi

1. Ürün Özelliklerinin Temsili: İlk adım, her bir ürünün özelliklerini sayısal veya kategorik degerler kullanarak temsil etmektir Bu özellikler arasında ürün kategorisi, marka, fiyat, renk, boyut, açıklama metni, kullanıcı değerlendirmeleri ve hatta görsel özellikler bulunabilir (Pazzani & Billsus, 2007).

2. Kullanıcı Profili Oluşturma: Kullanıcıların geçmişte beğendiği, satın aldığı veya etkileşimde bulunduğu ürünlerin özelliklerine dayanarak bir kullanıcı profili oluşturulur (Lops et al., 2011). Bu profil, kullanıcının ilgi alanlarını ve tercihlerini yansıtır.

3. Benzerlik Hesaplama: Kullanıcı profili ıle ürünlerin özellikleri arasındaki benzerlik hesaplanır Bu hesaplama için genellikle kosinüs benzerliği, Öklid uzaklığı veya Jaccard benzerliği gibi yöntemler kullanılır (Pazzani & Billsus, 2007).

4. Tavsiye Üretimi: Benzerlik skorlarına göre, kullanıcı profiline en çok benzeyen ürünler tavsiyelir. Bu tavsiyeler, kullanıcının ilgi alanlarına ve tercihlerine en uygun olan ürünlerdir (Pazzani & Billsus, 2007).

3.3.2. Avantajları

- **Kişiselleştirme:** Kullanıcıların bireysel tercihlerine ve ilgi alanlarına göre kişiselleştirilmiş tavsiyeler sunar
- **Yeni Ürün Tavsiyeleri:** Kullanıcıların geçmişte etkileşimde bulunmadığı ancak ilgi alanlarına uygun yeni ürünleri tavsiyer
- **Soğuk Başlangıç Problemi:** Yeni kullanıcılara veya yeni ürünlere tavsiyeler sunma konusunda daha başarılıdır (Pazzani & Billsus, 2007).
- **Şeffaflık:** Tavsiyelerin neden yapıldığı açık ve anlaşılabilirdir (Burke, 2002).

3.3.3. Dezavantajları

- **Sınırlı Keşif:** Kullanıcıların geçmişte beğendiği ürünlere benzer ürünler önerdiği için keşif imkanlarını sınırlayabilir.
- **Özellik Mühendisliği:** Ürün özelliklerinin doğru ve etkili bir şekilde temsil edilmesi önemlidir.
- **Veri Seyrekliği:** Kullanıcıların yeterli satış ve etkileşim verisi yoksa, tavsiyelerin kalitesi düşebilir.

3.3.4. Kullanım Alanları

- **E-ticaret Siteleri:** Ürün tavsiyeleri, kişiye özel kampanyalar ve içerik tavsiyeleri için kullanılır
- **Haber Siteleri:** Kullanıcıların ilgi alanlarına göre kişiselleştirilmiş haberler tavsiye eder.
- **Müzik ve Video Platformları:** Kullanıcıların dinleme veya izleme geçmişine göre kişiselleştirilmiş içerik tavsiyeler yapar.

3.4. İşbirlikçi Filtreleme

İşbirlikçi filtreleme (Collaborative Filtering – CF), e-ticaret sitelerinde kullanıcıların benzersiz tercihlerini ve ilgi alanlarını anlamak için "topluluk zekasından" yararlanan güçlü bir tavsiye sistemi yöntemidir (Ricci, Rokach, & Shapira, 2011). Bu yaklaşım, kullanıcıların geçmiş davranışlarını (satın almalar, derecelendirmeler, tıklamalar vb.) ve diğer kullanıcıların benzer davranışlarını analiz ederek kişiselleştirilmiş ürün tavsiyeleri sunar (Su & Khoshgoftaar, 2009). Örneğin, bir kullanıcı daha önce aksiyon filmlerini beğendiyse ve bu kullanıcıya benzer diğer kullanıcılar bilim kurgu filmlerini de beğendiyse, işbirlikçi filtreleme bu kullanıcıya bilim kurgu filmleri önerebilir.

İşbirlikçi filtreleme, e-ticarette kullanıcı deneyimini kişiselleştirme ve müşteri memnuniyetini artırmak için etkili bir yöntemdir. Kullanıcıların ilgi alanlarına ve tercihlerine göre özelleştirilmiş ürün tavsiyeleri sunarak, satışları artırmaya ve müşteri sadakatini güçlendirmeye yardımcı olabilir. Ancak, soğuk başlangıç problemi ve veri seyrekliği gibi zorluklarla başa çıkmak için diğer tavsiye sistemi yöntemleriyle birlikte kullanılması, örneğin içerik tabanlı filtreleme ile hibrit bir model oluşturulması, daha etkili sonuçlar verebilir (Burke, 2002).

3.4.1. Çalışma Prensibi

1. Veri Toplama: Kullanıcıların ürünlerle olan etkileşimlerini (satın almalar, derecelendirmeler, tıklamalar, sepete eklemeler, sayfada geçirilen süre vb.) içeren büyük bir veri seti toplanır (Su & Khoshgoftaar, 2009). Bu veriler, genellikle kullanıcı-ürün matrisi şeklinde düzenlenir, burada her satır bir kullanıcıyı, her sütun bir ürünü temsil eder ve hücrelerdeki değerler kullanıcıların ürünlerle olan etkileşimlerini gösterir.

2. Benzer Kullanıcıları veya Ürünleri Bulma: Kullanıcı tabanlı CF'de, odak kullanıcıya benzer davranışlar sergileyen diğer kullanıcılar belirlenir. Ürün tabanlı CF'de ise, hedef kullanıcının geçmişte beğendigi ürünlere benzer ürünler tespit edilir. Bu benzerlik hesaplaması için Pearson korelasyonu, kosinüs benzerliği veya Jaccard benzerliği gibi yöntemler kullanılabilir (Sarwar et al., 2001).

3. Tavsiye Üretimi: Kullanıcı tabanlı CF'de, benzer kullanıcıların beğendiği veya satın aldığı ancak hedef kullanıcının henüz etkileşimde bulunmadığı ürünler tavsiyelir (Breese, Heckerman, & Kadie, 1998). Ürün tabanlı CF'de ise, hedef kullanıcının geçmişte beğendiği ürünlere benzer olan ve henüz etkileşimde bulunmadığı ürünler tavsiyelir (Linden, Smith, & York, 2003). Model tabanlı CF'de ise, kullanıcı-ürün etkileşimlerini modelleme için makine öğrenmesi algoritmaları (matris çarpanlara ayırma, kümeleme vb.) kullanılarak tavsiyeler üretilir (Koren, Bell, & Volinsky, 2009).

3.4.2. İşbirlikçi Filtreleme Türleri

- **Kullanıcı Tabanlı (User-Based):** Hedef musterılere benzer kullanıcıların tercihlerine dayanarak tavsiyelerde bulunur.

- **Ürün Tabanlı (Item-Based):** Hedef kullanıcının geçmişte beğendiği ürünlere benzer ürünleri tavsiyer.

- **Model Tabanlı (Model-Based):** Kullanıcı-ürün etkileşimlerini modellemek için makine öğrenmesi algoritmaları (matris çarpanlara ayırma, kümeleme vb) kullanır.

3.4.3. Avantajları

- **Gizli Tercihleri Ortaya Çıkarma:** Kullanıcıların açıkça ifade etmedikleri ilgi alanlarını ve tercihlerini ortaya çıkarabilir.

- **Keşif İmkanı:** Kullanıcıların normalde karşılaşmayacağı ürünleri önererek keşif imkanı sunabilir.

- **Ölçeklenebilirlik:** Büyük veri setleri ve kullanıcı sayısı ile ölçelendirme yapabilir.

3.4.4. Dezavantajları

- **Soğuk Başlangıç Problemi:** Yeni kullanıcılara veya yeni ürünlere tavsiyeler sunma konusunda zorluk yaşayabilir (Lam, Vu, Le, & Duong, 2008).f

- **Veri Seyrekliği:** Kullanıcıların yeterli etkileşim verisi yoksa, tavsiyelerin kalitesi düşebilir.

- **Popülerliğe Yönelim:** Popüler ürünlere odaklanma eğiliminde olabilir.

3.4.5. Kullanım Alanları

- **E-ticaret Siteleri:** Ürün tavsiyeleri, kişiselleştirilmiş kampanyalar ve içerik tavsiyeleri için kullanılır.

- **Film ve Dizi Platformları:** Kullanıcıların izleme geçmişine göre kişiselleştirilmiş içerik

tavsiyer.

- **Müzik Platformları:** Kullanıcıların dinleme geçmişine göre kişiselleştirilmiş müzik tavsiyelinde bulunur.

4. TAVSİYE SİSTEMLERİNDE KARŞILAŞILAN PROBLEMLER

Tavsiye sistemleri, kullanıcı deneyimini kişiselleştirme ve kullanıcı memnuniyetini artırma potansiyeliyle günümüz dijital dunyasında önemli bir yere sahiptir. Ancak, bu sistemlerin etkili bir şekilde çalışması için çeşitli zorlukların atlatılması gerekmektedir. Bu bölümde, tavsiye sistemlerinde

karşılaşılan temel problemler, bunların nedenleri ve potansiyel çözüm yolları detaylı bir şekilde incelenecektir.

4.1. Soğuk Başlangıç Problemi

Tavsiye sistemlerinin en yaygın sorunlarından biri, yeni kullanıcılara veya yeni ürünlere yönelik tavsiyelerde bulunma güçlüğüdür. Yeni kullanıcılara ilişkin yeterli veri olmadığında, sistemin ozel tavsiyeler sunması güçleşir. Benzer şekilde, yeni ürünler için henüz yeterli kullanıcı etkileşimi olmadığından, bu ürünlerin tavsiyelmesi de zorlaşır (Schein et al., 2002).

Çözüm Yolları:

- **Hibrit Tavsiye Sistemleri:** İşbirlikçi filtreleme ve içerik tabanlı filtreleme gibi farklı yöntemleri birleştirerek, soğuk başlangıç probleminin etkisini azaltmak mümkündür (Burke, 2002).

- **Demografik Bilgileri Kullanma:** Kullanıcıların demografik bilgilerini (yaş, cinsiyet, konum vb.) kullanarak, benzer demografik özelliklere sahip kullanıcıların tercihlerine dayalı tavsiyelerde bulunabilir

- **Varsayılan Tavsiyeler:** Popüler veya trend olan ürünleri yeni kullanıcılara önermek, soğuk başlangıç problemini hafifletebilir.

4.2. Veri Seyrekliği

Birçok tavsiye sistemi, kullanıcı-ürün eşleşmesini içeren büyük ve seyrek matrisler üzerinde çalışır. Bu matrislerde, kullanıcıların çoğu ürünle etkileşimde bulunmadığı için çok sayıda boş hücre bulunur. Bu durum, tavsiyelerin doğruluğunu ve çeşitliliğini olumsuz etkileyebilir (Sarwar et al., 2001).

Çözüm Yolları:

- **Matris Çarpanlara Ayırma (Matrix Factorization):** Gizli faktör modelleri kullanarak, eksik değerleri tahmin etmek

ve veri seyrekliğini azaltmak mümkündür (Koren, Bell, & Volinsky, 2009).

- **Veri Zenginleştirme:** Kullanıcıların sosyal medya etkinlikleri, demografik bilgileri veya ürün meta verileri gibi ek verileri kullanarak, veri setini zenginleştirmek ve seyrekliği azaltmak mümkündür (Melville, Mooney, & Nagarajan, 2002).

- **Aktif Öğrenme:** Kullanıcılardan geri bildirim alarak, eksik değerleri doldurmak ve tavsiyelerin kalitesini artırmak mümkündür (Rashid et al., 2002).

4.3. Ölçeklenebilirlik

E-ticaret platformları gibi büyük ölçekli sistemlerde, milyonlarca kullanıcı ve ürün bulunabilir. Bu durumda, tavsiye sistemlerinin hesaplama maliyeti ve işlem süresi önemli bir sorun haline gelebilir (Linden, Smith, & York, 2003).

Çözüm Yolları:

- **Dağıtık Hesaplama:** Büyük veri işleme teknolojilerini (Hadoop, Spark vb.) kullanarak, tavsiye sistemlerini dağınık bir şekilde çalıştırmak ve ölçeklenebilirliği artırmak mümkündür (George & Merugu, 2005).

- **Yaklaşık Algoritmalar:** Kesin sonuçlar yerine yaklaşık sonuçlar üreten algoritmalar kullanarak, hesaplama maliyetini düşürmek mümkündür.

- **Önbellekleme (Caching):** Sık kullanılan veya popüler tavsiyeleri önbelleğe alarak, işlem süresini kısaltmak mümkündür.

4.4. Filtre Balonu

Tavsiye sistemleri, kullanıcıların geçmiş tercihlerine dayanarak tavsiyelerde bulunduğundan, kullanıcıları kendi ilgi alanlarına hapseden bir "filtre balonu" oluşturabilir. Bu durum,

kullanıcıların yeni ve farklı ürünler keşfetmelerini engelleyebilir (Ekstrand, Riedl, & Konstan, 2011).

Çözüm Yolları:

- **Çeşitlilik Odaklı Tavsiyeler:** Popülerlik veya benzerlik yerine çeşitlilik ölçütlerini kullanarak, kullanıcılara farklı kategorilerden veya ilgi alanlarından ürünler önermek faydalı olabilir.

- **Rastgele Öneri:** Rastgele veya beklenmedik tavsiyeler sunarak, kullanıcıların yeni ve farklı ürünler keşfetmelerini teşvik etmek mümkündür.

- **Açıklama Sağlama:** Tavsiyelerin neden yapıldığını açıklayarak, kullanıcıların filtre balonunun farkında olmalarını ve farklı seçenekleri keşfetmelerini sağlamak mümkündür.

4.5. Gizlilik ve Güven

Tavsiye sistemleri, kullanıcıların kişisel verilerini ve davranışlarını analiz ettiği için gizlilik ve güven endişelerine sebebiyet verebilir.

Çözüm Yolları:

- **Anonimleştirme:** Kullanıcı verilerini anonimleştirerek, kişisel bilgilerin gizliliğini korumak mümkündür.

- **Şeffaflık:** Tavsiye sistemlerinin nasıl çalıştığını ve hangi verileri kullandığını açıklayarak, kullanıcıların güvenini kazanılabilir.

- **Kullanıcı Kontrolü:** Kullanıcıların hangi verilerin kullanılacağını ve hangi tavsiyeleri görmek istediklerini kontrol etmelerine izin vermek faydalı olabilir

5. UYGULAMA

E-ticaret sitelerinde tavsiye sistemlerini anlamak amacı ile bu bölümde 2 farklı e-ticaret sitesinden edinilmiş veri setleri ile 3 farklı tavsiye geliştirilecektir. Bu tavsiye sistemleri; Basit tavsiye sistemi, işbirlikçi tavsiye sitemi ve içerik tabanlı filtreleme sistemi olacaktır.

5.1. Veri Seti Hikayeleri ve Değişkenleri

5.1.1. Veri Seti 1: Amazon Ürün Değerlendirmesi Veri Seti

Bu veri setinde, Amazon E-ticaret platformunda müşterilerin elektronik ürün satın alımlarında ürünlere verdikleri puanlar toplanmıştır. Veri seti, müşteri kodu, ürün kodu, değerlendirme puanları ve değerlendirme zamanı bilgilerini içermektedir. Veriler, 2023 yılındaki alışverişlerden veri madenciliği yöntemleri ile toplanmış ve aşağıdaki kaynakta bir araya getirilmiştir.

Kaynak: https://amazon-reviews-2023.github.io/

Değişkenler:

- **userId:** Müşteri kodu (ilk sütun)
- **productId:** Ürün kodu (ikinci sütun)
- **Rating:** Müşterinin değerlendirmesi (1 - 5) (üçüncü sütun)
- **timestamp:** Değerlendirme zamanı (dördüncü sütun)

5.1.2. Veri Seti 2: Home Depo Ürün Açıklamaları Veri Seti

Bu veri seti, Home Depot web sitesindeki ürünlerin ve detaylarının bir listesini içermektedir. Veri seti, firma tarafından açık kaynak yoluyla ürün tavsiye sistemlerinin geliştirilmesi amacıyla aşağıda belirtilen kaynakta paylaşılmıştır.

Kaynak:https://www.kaggle.com/competitions/home-depot-product-search-relevance/data?select=product_descriptions.csv.zip

Değişkenler:

- **product_uid:** Ürün kodu
- **product_description:** Ürün açıklamaları

5.2. Basit Tavsiye Sistemi Uygulaması

Basit tavsiye sistemlerinden olan popülariteye dayalı bir analiz yapılarak web sitesinde en çok satılan ürünleri yeni müşterilere sunmak için bir analiz yapılacaktır.

Şekil 1.1 "de analiz için gerekli indirilen kütüphaneler yer almaktadır.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use("ggplot")
import sklearn
from sklearn.decomposition import TruncatedSVD
```

Şekil 1.2"de veri seti okunmuş ve kolon isimleri düzeltilmiş ve ön işleme yapılıp kayıp veri olan satırlar silinmiştir.

Şekil 1.2. Veri Setinin Okunması ve Veri Seti Ön İşleme Yapılması

```
data = pd.read_csv(r"/content/Amazon_Product_Reviews.csv", header = None)
data.columns = ['user_id', 'product_id', 'rating', 'timestamp']
data = data.dropna()
data.drop(['timestamp'], axis =1, inplace = True)
```

Şekil 1.3. "de veri seti içerisinden 5 örnek görselleştirilmiştir.

Şekil 1.3. Veri Seti İçerisinden Örnek

```
popular_products = pd.DataFrame(data.groupby('product_id')['rating'].count())
most_popular = popular_products.sort_values('rating', ascending=False)
most_popular.head(30).plot(kind = "bar")
```

Şekil 1.4. "de veri setindeki en çok değerlendirme alan; popular ürünler seçilmiş ve görselleştirilmiştir.

Şekil 1.4. Veri Seti İçerisinden Popülerleri Seçmek

	user_id	product_id	rating
0	AKM1MP6P0OYPR	0132793040	5.0
1	A2CX7LUOHB2NDG	0321732944	5.0
2	A2NWSAGRHCP8N5	0439886341	1.0
3	A2WNBOD3WNDNKT	0439886341	3.0
4	A1GI0U4ZRJA8WN	0439886341	1.0

5.3. İşbirlikçi Filtreleme Uygulaması

Bu yaklaşımda, kullanıcıların satın alma geçmişleri ve diğer kullanıcıların verdiği benzer puanlar dikkate alınarak ürün tavsiyeleri yapılır. Model tabanlı işbirlikçi filtreleme tekniği, birden fazla kullanıcının verilerindeki tercihlere dayalı kalıpları belirleyerek belirli bir kullanıcı için ürün tahminleri yapmaya yardımcı olduğu için burada tercih edilir.

Kullanım Matrisi (Utility Matrix): tavsiye sistemlerinde kullanılan bir kavramdır. Kullanıcıların ürünlere verdikleri puanları veya tercihlerini temsil eden bir matristir. Matrisin satırları kullanıcıları, sütunları ise ürünleri temsil eder. Matrisin içindeki her bir hücre, ilgili kullanıcının ilgili ürüne verdiği puanı veya tercihini gösterir.Bu uygulama da da kullanıcı ve ürün tercihlerini bu matris yöntemi ile ele alacağız.

Şekil 1.6'da kullanıcı matrisi oluştururken analiz yapabilmek adına ilk 10000 girdiyi seçip matrisi oluşturuyoruz.

Şekil 1.6. Kullanım Matrisi Oluşturulması

user_id	A00766851QZZUBOVF4JFT	A01255851ZO1U93P8RKGE	A0293130VTX2ZXA70JQS
product_id			
0132793040	0	0	0
0321732944	0	0	0
0439886341	0	0	0

Şekil 1.7. Matris Görseli

```
SVD = TruncatedSVD(n_components=10)
decomposed_matrix = SVD.fit_transform(X)
decomposed_matrix.shape
correlation_matrix = np.corrcoef(decomposed_matrix)
correlation_matrix.shape
```

Şekil 1.8. görselinde tekil değer ayrıştırması (SVD) yöntemi kullanıldı ve büyük boyuttaki veri kümeleri için baskın özellikleri çıkarmak ve boyut azaltmak için bu yöntem tercih edildi.

Şekil 1.8. Korelasyon matrisi oluşturulması ve modelin kurulması Şekil 1.8."de rasgele bi ürün seçiliyor ve index numarası teyit ediliyor.

Şekil 1.9. Rastgele Bir Ürün İndeksinin Seçilmesi

```
X.index[100]  # 1616833742
i = "1616833742"
product_names = list(X.index)
product_ID = product_names.index(i)
product_ID
```

Şekil 1.9.”de korelasyonu 0.90 “dan büyük olan ürünler listelenmiş ve rastgele seçilen ürün listeden çıkarılmış ve korelasyonu en büyük ve seçtiğimiz ürünü alanlara tavsiye edeceğimiz ilk 10 ürünü belirliyoruz.

Şekil 1.10. Tavsiye Edilen Ürünlerin Belirlenmesi

```
correlation_product_ID = correlation_matrix[product_ID]
Recommend = list(X.index[correlation_product_ID > 0.90])
Recommend.remove(i)
Recommend[0:9]
```

Şekil 1.10.”da rastgele seçilen “1616833742” kadlu ürünü alanlara aşağıda kodları bulunan 10 ürünü tavsiye etmemiz gerektiği bilgisini ediniyoruz.

Şekil 1.11. Tavsiye Edilen Ürünler

```
['1182702627',
 '1600775160',
 '1615598774',
 '1903874092',
 '1932836306',
 '3778939742',
 '3993854705',
 '7507825604',
 '7798902724']
```

5.4. İçerik Tabanlı Filtreleme Uygulaması

Kullanıcı-ürün satın alma geçmişi olmayan bir işletme için, kullanıcılara yönelik bir arama motoruna dayalı tavsiye sistemi tasarlanabilir. Ürün tavsiyeleri, ürün açıklamasında verilen metinsel kümeleme analizine dayalı olarak yapılabilir. Belirlenen veri setinde ürünlerin açıklamalarını kullanarak anahtar kelimeler üzerinden ürün/kategori tavsiyesi üzerine çalışılacaktır.

Şekil 1.12. Gerekli kütüphanelerin indirilmesi

```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.neighbors import NearestNeighbors
from sklearn.cluster import KMeans
from sklearn.metrics import adjusted_rand_score
```

Şekil 1.12.”de veri okunmuş, boş satırlar silinmiş ve veri setinin en üstündeki 10 satır görselleştirilmiştir.

Şekil 1.13. Verini Okunması ve İlk Ön İşlemelerin Yapılması

```
product_descriptions = pd.read_csv('/content/product_descriptions.csv' )
product_descriptions = product_descriptions.dropna()
product_descriptions.shape
product_descriptions.head()
```

Şekil 1.14. Örnek Verilerin Gösterilmesi

	product_uid	product_description
0	100001	Not only do angles make joints stronger, they ...
1	100002	BEHR Premium Textured DECKOVER is an innovativ...
2	100003	Classic architecture meets contemporary design...
3	100004	The Grape Solar 265-Watt Polycrystalline PV So...
4	100005	Update your bathroom with the Delta Vero Singl...

Şekil 1.14.’de Ürün tanımlarında yer alan İngilizce kelimeler vektörize edilip her biri sayısal degree dönüştürülmüştür

Şekil 1.15. Ürün Tanımlarındaki Kelimeleri Vektörize Edilmesi

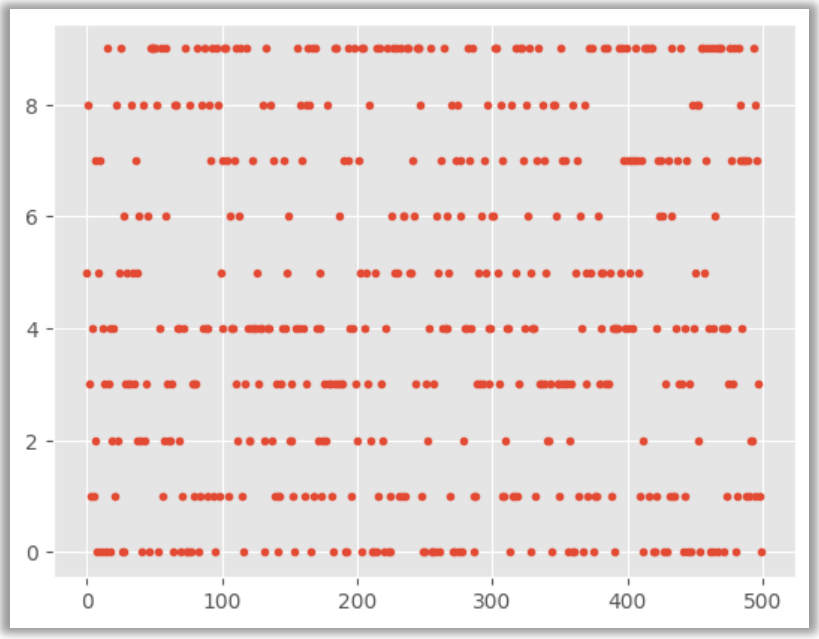
```
product_descriptions1 = product_descriptions.head(500)
product_descriptions1["product_description"].head(10)
vectorizer = TfidfVectorizer(stop_words='english')
X1 = vectorizer.fit_transform(product_descriptions1["product_description"])
X1
```

Şekil 1.15.'de K-mean kümeleme algoritmasını veri setine entegre ediyoruz ve grafik ile görselleştiriyoruz.

Şekil 1.16. K-Mean Kümeleme Algoritması

```
X=X1
kmeans = KMeans(n_clusters = 10, init = 'k-means++')
y_kmeans = kmeans.fit_predict(X)
plt.plot(y_kmeans, ".")
plt.show()
```

Şekil 1.17. Vektörlerin Görselleştirilmesi



Şekil 1.17.de Kullanıcının seçtiği ürüne dayalı olarak ürün tavsiyesi sistemi kurulmuştur. Sıkça birlikte satın alınanlara dayalı olarak ilgili ürünleri önermek temel hedeftir.

Şekil 1.18. Kümelerin Belirlenmesi

```
def print_cluster(i):
    print("Cluster %d:" % i),
    for ind in order_centroids[i, :10]:
        print(' %s' % terms[ind]),
    print

true_k = 10

model = KMeans(n_clusters=true_k, init='k-means++', max_iter=100, n_init=1)
model.fit(X1)

print("Kümeler:")
order_centroids = model.cluster_centers_.argsort()[:, ::-1]
terms = vectorizer.get_feature_names_out()
for i in range(true_k):
    print_cluster(i)
```

Şekil 1.19. Örnek Kümeler

```
Kümeler:
Cluster 0:
 air
 light
 power
 control
 water
 ft
 window
 fan
 energy
 easy
Cluster 1:
 metal
 projects
 screw
 gauge
 used
```

Şekil 1.19.'de Tavsiye fonksiyonlaştırılmış ve örnek olarak fonksiyona "water" kelimesi verilmiştir.

4. SONUÇ

Bu çalışma, e-ticaret siteleri için ürün tavsiye sistemlerinin önemini ve potansiyelini vurgulamaktadır. Kullanıcıların tercihlerine ve davranışlarına göre kişiselleştirilmiş ürün tavsiyeleri sunarak müşteri memnuniyetini artırmak ve satışları yükseltmek için bu sistemlerin nasıl kullanılabileceğini göstermektedir.

Çalışmada, basit tavsiye sistemleri, birliktelik kuralı öğrenimi, içerik tabanlı filtreleme ve işbirlikçi filtreleme gibi farklı yöntemlerin avantajları ve dezavantajları ele alınmıştır. Her yöntemin farklı kullanım senaryoları ve veri gereksinimleri olduğu belirtilmiştir.

Ayrıca, tavsiye sistemlerinin karşılaştığı soğuk başlangıç problemi, veri seyrekliği, ölçeklenebilirlik, filtre balonu ve gizlilik gibi zorluklar da tartışılmıştır. Bu zorlukların üstesinden gelmek için hibrit modeller, veri zenginleştirme, dağıtık hesaplama, çeşitlilik odaklı tavsiyeler, anonimleştirme ve şeffaflık gibi çözüm yolları tavsiyelmiştir.

Sonuç olarak, bu çalışma, e-ticaret siteleri için ürün tavsiye sistemlerinin karmaşıklığını ve çeşitliliğini ortaya koymaktadır. İşletmelerin, kendi özel ihtiyaçlarına ve hedeflerine uygun olan en iyi yöntemi seçmek için farklı yöntemleri dikkatlice değerlendirmeleri gerekmektedir.

KAYNAKÇA

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB (Vol. 1215, pp. 487-499).

Breese, J. S., Heckerman, D., & Kadie, C. (1998, August). Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence (pp. 43-52). Morgan Kaufmann Publishers Inc.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. User modeling and user-adapted interaction, 12(4), 331-370.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., & Sartin, M. (1999, April). Combining content-based and collaborative filters in an online newspaper. In Proceedings of the ACM SIGIR workshop on recommender systems (pp. 6-10).

Davidson, J., Liebald, B., Liu, J., Nandy, P., Van Vleet, T., Gargi, U., ... & Zhou, Y. (2010). The youtube video recommendation system. In Proceedings of the fourth ACM conference on Recommender systems (pp. 293-296).

Ekstrand, M. D., Riedl, J. T., & Konstan, J. A. (2011). Collaborative filtering recommender systems. Foundations and Trends® in Human–Computer Interaction, 4(2), 81-173.

Hahsler, M., Grün, B., & Hornik, K. (2005). arules - A computational environment for mining association rules and frequent item sets. Journal of Statistical Software, 14(15), 1-25.

Han, J., Pei, J., & Yin, Y. (2000, April). Mining frequent patterns without candidate generation. In ACM sigmod record (Vol. 29, No. 2, pp. 1-12). ACM.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37.

Kültür, Y., & Özmen, A. (2016). Perakende sektöründe birliktelik kuralları analizi uygulaması: Bir süpermarket örneği. İşletme Araştırmaları Dergisi, 8(3), 127-142.

Lam, X. N., Vu, T., Le, T. D., & Duong, A. D. (2008). Addressing cold-start problem in recommendation systems. In Proceedings of the 2nd international conference on Ubiquitous information management and communication (pp. 208-211).

Lin, W., Alvarez, S. A., & Ruiz, C. (2002). Efficient adaptive-support association rule mining for recommender systems. Data mining and knowledge discovery, 6(1), 83-105.

Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. IEEE Internet computing, 7(1), 76-80.

Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In Recommender systems handbook (pp. 73-105). Springer, Boston, MA.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval. Cambridge university press.

Melville, P., Mooney, R. J., & Nagarajan, R. (2002, July). Content-boosted collaborative filtering for improved recommendations. In Aaai/iaai (pp. 187-192).

Pazzani, M. J., & Billsus, D. (2007). Content-based recommendation systems. In The adaptive web (pp. 325-341). Springer, Berlin, Heidelberg.

Rashid, A. M., Albert, I., Cosley, D., Lam, S. K., McNee, S. M., Konstan, J. A., & Riedl, J. (2002, July). Getting to know you: learning new user preferences in recommender systems. In Proceedings of the 7th international conference on Intelligent user interfaces (pp. 127-134).

Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to recommender systems handbook. Springer US.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web (pp. 285-295).

Schafer, J. B., Konstan, J., & Riedl, J. (1999, November). Recommender systems in e-commerce. In Proceedings of the 1st ACM conference on Electronic commerce (pp. 158-166).

Schedl, M., Zamani, H., Chen, C., Deldjoo, Y., & Elahi, M. (2018). Current challenges and visions in music recommender systems research. International Journal of Multimedia Information Retrieval, 7(2), 95-116.

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2002, July). Methods and metrics for cold-start recommendations. In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 253-260).

Şengör, İ. (2019). Market sepeti analizinde birliktelik kuralları ve veri madenciliği teknikleri. İstanbul Üniversitesi İşletme Fakültesi Dergisi, 48(2), 214-227.

Smith, B., & Linden, G. (2017). Two decades of recommender systems at amazon. com. IEEE Internet Computing, 21(3), 12-18.

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. Advances in artificial intelligence, 2009.

Tan, P. N., Steinbach, M., & Kumar, V. (2005). Introduction to data mining. Pearson Education India.

Zaki, M. J. (2000). Scalable algorithms for association mining. IEEE transactions on knowledge and data engineering, 12(3), 372-390.

Zhang, Y., Jin, R., & Zhou, Z. H. (2014). Understanding bag-of-words model: a statistical framework. International Journal of Machine Learning and Cybernetics, 1(1-4), 43-52.