Araştırma Makalesi / Research Paper
Bayburt Üniversitesi Fen Bilimleri Dergisi
Bayburt University Journal of Science

dergipark.gov.tr/bufbd
ISSN: 2667-579X

Cilt 5, Sayı 2, Yıl 2022, Sayfa 173-188
Vol 5, Issue 2, Year 2022, Pages 173-188
doi.org/bufbd.1119062

Geliş Tarihi / Date of Recieve: 20.05.2022
Revizyon Tarihi / Date of Revision: 06.06.2022
Kabul Tarihi / Date of Acceptance: 06.06.2022

Bir Perakende Firmasında Metin Benzerliği ve Tekil Değer Ayrışımı Algoritması Tabanlı Ürün Öneri Sisteminin Oluşturulması

Alper KİRAZ*, Bilal ERDEMİR

Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Endüstri Mühendisliği, Sakarya, Türkiye

Anahtar Kelimeler:

Ürün öneri sistemi,
E-Ticaret,
Metin benzerliği,
Tekil değer ayrışımı

Özet

Günümüzde şirketlerin dijital dönüşüm kapsamında yaptığı çalışmalar özellikle pandemi sonrası tüketiciden gelen talebin artmasıyla giderek hız kazanmıştır. Bu bağlamda, E-Ticaret alanında web site ve mobil uygulamalarda müşteriye en uygun ürün önerilerinin sunulması, müşteri ihtiyaçlarının karşılanması ve şirketlerin satış hedeflerinin gerçekleştirilmesi için ürün öneri sistemlerine ihtiyaç duyulmuştur.
Bu alanda yapılan çalışmalar, ürün tipi ve çeşitliliği değişkenlik gösterdiği için çoğu zaman mantıksız ve yanlış ürün önerilerinin tüketicilere sunulmasına yol açtığı, farklı sitelerin ürün önerileri incelendiğinde açıkça görülmüştür. Bu çalışmada, şirketin ürün ve veri yapısına göre en uygun şekilde veri manipülasyonun gerçekleştirilmesi, özelleştirilmiş fonksiyonların yazılması, Metin Benzerliği ve Tekil Değer Ayrışımı
algoritmasına dahil olarak en uygun tamamlayıcı ürünlerin gösterilmesi sağlanmıştır. En uygun algoritmanın geliştirilerek yazılımı tamamlanmıştır. Elde edilen ürün önerileri web sitesi ve mobil uygulamada canlıya alınmıştır. Sonuçlar Google Analytics üzerinden ve Python kodlarıyla ayrı ayrı gözlemlenmiştir.
Yapılan çalışma sonucunda geliştirilen ürün öneri sisteminin mevcut sisteme kıyasla set halinde satılan ürünlerde %7,62, tekli olarak satılan ürünlerde ise %11,2 oranında daha iyi performans gösterdiği gözlemlenmiştir. Ayrıca ilgili alanın web site ve mobil uygulamada görüntülenme sayısı %7,17, tıklanma sayısı %28,93 artış göstermiştir. Ürün önerilerinin mevcut duruma kıyasla daha mantıklı ve tamamlayıcı olarak daha ilişkili ürün önerileri sunduğu tespit edilmiştir ve ilgili alanlar farklı zamanlarda incelenerek gözlemlenmiştir.

*e-posta: erdemirbll@gmail.com

Bu makaleye atıf yapmak için:
Alper KİRAZ; Bilal ERDEMİR, "Bir Perakende Firmasında Metin Benzerliği ve Tekil Değer Ayrışımı Algoritması Tabanlı Ürün Öneri Sisteminin Oluşturulması", Bayburt Üniversitesi Fen Bilimleri Dergisi, C. 5, s 2, ss. 173-188

How to cite this article:
Alper KİRAZ; Bilal ERDEMİR, "Creating a Text Similarity and Singular Value Decomposition Algorithm Based Product Recommendation System in a Retail Company", Bayburt University Journal of Science, vol. 5, no 2, pp. 173-188

Creating a Text Similarity and Singular Value Decomposition Algorithm Based Product Recommendation System in a Retail Company

Keywords:

Product recommendation engine, E-Commerce Text similarity, Singular value decomposition

Abstract

Due to rising client demand following the pandemic, organizations' initiatives in the field of digital transformation are gaining pace nowadays. Product recommendation systems were necessary for this context to deliver the most relevant product choices to clients, meet their expectations, and accomplish the businesses' sales targets through the e-Commerce website and mobile applications.

Examining the product suggestions of numerous websites reveals that study on this subject frequently results in the presentation of illogical and erroneous product recommendations to visitors, given the variety and type of things accessible. This study looked into data manipulation, building custom functions, Text Similarity, and Singular Value Decomposition. The algorithm displays the best complementary commodities. The software was completed by developing the best algorithm. The product proposals were brought to life through the use of a website and a mobile application. Individual observations were made using Google Analytics and Python scripts.

The study found that the created product recommendation system beat the current system by 7.62 percent in items sold as a group and by 11.2 percent in products sold separately. Furthermore, the number of views on the website and mobile application for the relevant area climbed by 7.17 percent, and the number of clicks increased by 28.93 percent. It has been decided that the product suggestions provide more logical and complementary product recommendations than the existing situation, and the relevant regions have been monitored and reviewed at various times.

1 GİRİŞ

Günümüzde tüketim ihtiyaçlarının artmasına yönelik olarak perakende sektöründe faaliyet gösteren firmalar, ürün çeşitliliğini ve faaliyet alanlarını genişleterek müşteriye daha çok alternatif sunmak ve satış hedeflerini yükseltmek istemektedir. Özellikle pandemi sonrası tüketici davranışlarının dünyada e-ticaret alanına yönelmesiyle birlikte firmalar gelişen teknolojik gelişmeler neticesinde dijital dönüşüm kapsamında yatırımlarına ağırlık vermektedir. Bu alanda en öne çıkan konulardan birini de ürün öneri sistemleri oluşturmaktadır.

İnternetin her geçen gün yaygınlaşmasıyla kullanıcılara ait verilerden oluşan büyük veri ambarlarından değerli bilgilerin çıkarılması için analiz çalışmaları yapılmış ve öneri sistemleri 1990'lı yılların ortalarından itibaren veri teknolojileri içerisinde farklı bir başlık olarak incelenmeye başlanmıştır [1]. Öneri sistemleri e-ticaret sitelerinden satın alınacak bir ürün, sosyal medya kanallarında izlenecek bir video, film, dizi, haber, kitap veya müzik gibi çok farklı alanlarda kullanıcıların ihtiyaçlarını karşılamak için zaman ve fayda sağlamayı hedefleyen, belirlenmiş kullanıcıların en çok ilgisini çekebilecek nesnelerin önerilmesini sağlayan yazılım araçları ve tekinikleridir [2]. Bu bağlamda öneri sistemlerinin en yaygın olarak kullanıldığı ve üzerinde yoğun çalışmaların yapıldığı Netflix [3], Amazon [4], Youtube [5] ve Spotify [6] gibi platformlar öne çıkmaktadır. Bu platformlar çok sayıda kullanıcıdan beslenen büyük veri yapılarıyla çalıştığından dolayı başarı oranının daha yüksek olarak gerçekleştiği bilinmektedir. Ürün satışı yapmak isteyen çoğu perakende firmalarında ise veri büyüklüğünün az olması, ürün çeşitliliğinin artması ve firma yapılarından kaynaklı olarak ürün öneri sistemlerinin mantıksız ve ilgili olmayan ürün önerileri sunduğu gözlemlenmiştir.

Öneri sistemleri siteye trafik çekmek, müşterilere ilgili ve tamamlayıcı ürünlerin gösterimini gerçekleştirmek, satışların ve ürün görüntülenme sayılarının artmasını sağlamak, kullanıcılara en uygun kişiselleştirilmiş seçenekleri sunarak karar vermeyi kolaylaştırmak gibi avantajları sunmaktadır. Buna karşın ürün veya hizmet önerilerinin yapılırken bulunduğu sektör ve ilgi alanına göre değişen dinamiklere uyum sağlayamama, soğuk başlangıç, veri yetersizliği ve seyrek olma durumu, yeterli sürede çalışamama gibi dezavantajları da bulunmaktadır.

Bu kapsamda öneri sistemleri ile ilgili yapılan çalışmalar incelenmiştir. Genel olarak farklı algoritmaların performans değerleri karşılaştırılarak birinin seçilmesi ve bu doğrultuda ürün önerilerinin gerçekleştirildiği

gözlemlenmiştir. Uygulandığı sektör, algoritma tercihleri ve veri yapılarının uygunluğu göz önünde bulundurularak hangi yöntemlerin daha iyi sonuç verdiği araştırılmıştır. Özellikle ürün çeşitliliğinin son derece fazla ve değişken olduğu züccaciye sektörüne ait bir çalışmaya rastlanmamıştır. Metin benzerliği ve Tekil Değer Ayrışımı algoritması yaklaşımının birlikte kullanıldığı herhangi bir ürün öneri sistemi de literatürde yer almamaktadır. Bu bağlamda tercih edilen metotlar ve uygulandığı sektör bakımından, geliştirilen ürün öneri sisteminin özgünlüğü ortaya konulmuştur.

Bozkurt, Acı (2021) MovieLens veri seti kullanılarak film önerilerinin tekil değer ayrışımı, işbirlikçi filtreleme ve kosinüs benzerliği yöntemleri kullanarak 3 farklı metodun karşılaştırılması yapılmıştır. Kullanıcılara öneriler hakkında düşünceler sorulduğunda en iyi sonuçlar tekil değer ayrışımı ile elde edilirken yapılan çalışma sonucunda işbirlikçi filtrelemenin daha başarılı olduğu gözlemlenmiştir [7]. İncelenen bu çalışma neticesinde TDA'nın avantajları açıkça görülmüştür ve geliştirilen ürün öneri sistemi için korelasyon değerlerinin önemi anlaşılmıştır. Yalçın (2021), Ürün profilleri ayrı bir yaklaşımla, beğenilme dereceleriyle incelenmiştir. İşbirlikçi filtreleme algoritmalarının en çok tercih edilen ürünlere yönelik yanlılık durumu araştırılmıştır. 3 farklı kategoriden 9 önemli işbirlikçi filtreleme algoritmasını kullanarak iki reel dünya veri kümesi üzerinde farklı deneyler gerçekleştirilmiş. Yapılan deneyler sonucunda neredeyse bütün algoritmaların çok beğenilen ürünlere yanlılığın yüksek olduğunu ve kaliteli öneriler üretmede TDA, TDA++ gibi matris çarpanlarına ayırma tabanlı algoritmaların başarılı olduğu tespit edilmiştir [8]. İncelenen bu çalışmada, Tekil Değer Ayrışımının algoritmalar arasında en iyi ürün önerileri sunduğu gözlemlenmiştir. Geliştirilen ürün öneri sisteminde müşteriler satın aldığı ürünü beğenmiştir kabulü ile ilerlenmiştir. Bu açıdan yapılan çalışmaya katkıda bulunmuştur. Dundar, Görgülü Kalkisim (2021), Çalışmalarında giyim metaverilerine dayalı temsil öğrenimi ile kıyafet öneri sistemi geliştirmiştir. Rastgele yürüyüş ve Skipgram yöntemlerini kullanarak ürünler arası benzerliği modelleyen metaveriler üzerinden yeni bir yaklaşımda bulunmuştur. Yöntemin sınıflandırma başarı performansını ölçmek için Rastgele Orman (RF) kullanılmış olup F-ölçüm (Mikro) %76,72 bulunmuştur. Karşılaştırma için kullanılan yöntemlerde F-ölçüm (Mikro) değerleri Word2Vec için %62,92 ve Doc2Vec için %60,74 olarak sınıflandırma başarısı göstermiştir [9]. Perakende sektöründe giyim alanında yapılan bu çalışma incelenerek özellikle ev tekstili kategorisinde yapılacak ürün önerilerine katkı sağlanmıştır. Choi, Kim (2014) Çalışmalarında e-ticaret siteleri içinde müşterilere ait tekrarlı satın alma işlemlerine yönelik bir öneri sistemi geliştirmişlerdir. Kullanıcılardan elde edilen veriler geliştirilen öneri sisteminde kullanılmıştır. Veri setinde yaklaşık 5000 kullanıcı ve 4000 ürün kullanılarak öneri sisteminde performans analizleri yapılmıştır. Sonuç olarak kullanıcı tabanlı filtreleme tekniğinin ürün tabanlı filtreleme tekniğinden daha iyi performans gösterdiği tespit edilmiştir [10]. Çalışma incelenerek ilk olarak kullanıcı tabanlı ilerlenmesi planlanmıştır. Burada kullanıcılara ait büyük bir veri üzerinden hesaplamalarda eksiklikler tespit edildiği için ürün içeriklerini baz alarak ilerleme kararı alınmıştır. Alharthi ve ark. 2017 yılında kitap öneri sistemlerine ilişkin bir literatür çalışması yapmıştır. Literatürde kullanılan veri setleri ile öneri sistemleri yöntemlerini araştırmıştır. Yapılan çalışmada işbirlikçi filtreleme ve içerik tabanlı yöntemler ön plana çıkmıştır. INEX Book Track, Book-Crossing veriseti, LibraryThing, LitRec, Amazon reviews, Goodreads ve Project Gutenberg gibi veri setlerinin kitap öneri sistemi için ön plana çıktığını tespit etmiş bu veriler hakkında bilgilere değinilmiştir [11]. Güler (2019), Çalışmasında öneri sistemlerine genel bir bakış açısıyla yaklaşmıştır. Öneri sistemlerinde kullanılan yöntemler, karşılaşılan problemler ve bu sistemlerin uygulamaları ve karşılaştırmaları üzerinde durmuştur. Öneri sistemleri ile ilgili araştırma yapan kullanıcılar için etkileşimlerde ilk 10'a girme oranlarını hibrit yöntem %53, içerik bazlı yöntem %52, işbirlikçi yöntem %45 olarak tespit ederken ilk 5'e girme oranlarını hibrit yöntem %43, içerik bazlı yöntem 41 ve işbirlikçi yöntem için %33 olarak hesaplamıştır [12]. Bu çalışmanın incelenmesi neticesinde hibrit ve işbirlikçi yöntemlerin birbirine çok yakın sonuç değerlerine sahip olduğu gözlenmiştir ve bir sonraki çalışmada hibrit yöntem ile de karşılaştırmalı sonuç değerlerine ulaşılması planlanmıştır.

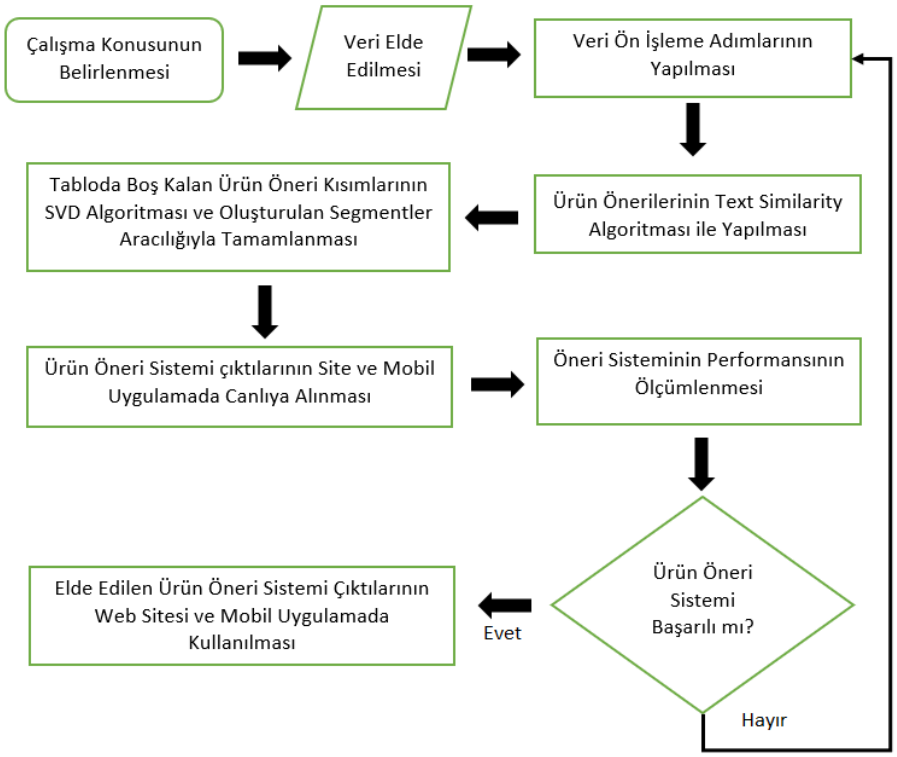
Uçar (2021), E-ticaret sitelerine yönelik makine öğrenmesi ve bulanık tabanlı bir öneri sistemi geliştirmiştir. Farklı sitelerde görüntülenen ve satın alınan kitap verileri ile desteksiz öğrenme yöntemiyle kümeleme yapılmıştır. Alt kümeleri K-Ortalamalar algoritması kullanarak kümeleme işlemlerinden elde etmiştir. Ürün önerilerinin geliştirmesinde bulanık model uygulamanın avantajlarını ve bu modelin kurallarının belirlenmesinde karar ağacı metodunun önemini göstermiştir [13]. Yıldız(2018), Moda perakendesi alanında müşteri segmentasyonu temelli bir ürün öneri sistemi geliştirmiştir. RFM analizi ile edilen skorlar ile müşteri bilgileri birleştirilerek k-ortalamalar tekniğiyle kümeleme işlemi yapılmıştır. Elde edilen 3 kümeye birliktelik analizi yapılarak hangi kategorileri birlikte aldığı tespit edilmiş ve ürün önerileri yapılmıştır. Geliştirilen sistemin mevcut sisteme göre hem f1 hem de ortalama satış değeri bakımından pozitif etkisi gözlemlenmiştir [14]. Çalışmalar göstermiştir ki fiyat segmentasyonu oluşturmada k-ortalamalar tekniğinin kullanılarak kümeleme işlemlerin yapılması öneri sisteminde amaçlanan sonuçlara ulaşmak için en uygun yöntemdir [13-14].

Geliştirilen ürün öneri sisteminde algoritmaların firmaya ait mevcut veri yapılarına uygunluğu göz önünde bulundurulmuştur ve ürün önerileri yapılırken öncelikli olarak metin benzerliği ve tekil değer ayrışımına ek olarak k-ortalamalar, özelleştirilmiş fonksiyonlar gibi çok yönlü bir ürün öneri sistemi geliştirilmiştir.

Yapılan çalışmanın amacı, perakende sektöründe faaliyet gösteren bir firmaya ait mevcut durumda bulunan ürün öneri sisteminde hatalı, eksik ve ilgili olmayan ürün önerilerinin gözlemlenmesi sonucu geliştirilerek firmanın satışlarını artırmak, kullanıcılara en uygun tamamlayıcı ürünleri sunmak ve buna bağlı olarak müşteri memnuniyetini sağlamaktır. Perakende sektöründe önemli bir yer tutan züccaciye alanında yapılmış olan ilk çalışma niteliğini taşıyarak bu alanda yapılacak çalışmalara öncülük etmesi hedeflenmiştir. Ayrıca kullanıcılara canlı bir sistemde gösterimi ve geri bildirimlere göre düzenlenebilir esnek bir yapı kazandırılarak sürdürülebilir olması sağlanmıştır. Ayrıca canlı sistemde müşteriye doğrudan temas eden bir yapıya talep edilen veya eksik görülen her alanda değişiklik yapılabilecek esnek bir yapı kazandırılmıştır.

2 MATERYAL VE METODOLOJİ

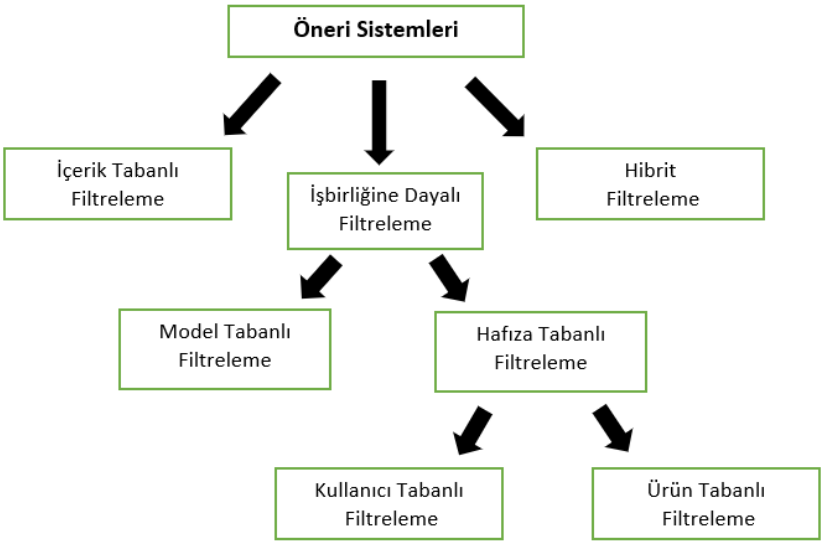
Çalışmada züccaciye, elektrikli ev aletleri, dekoratif ürünler ve ev tekstili alanında satış faaliyeti gösteren bir firmaya ait 8 aylık veriler baz alınmıştır. Ürünlerin stokta bulunma ve aktif satışta olmasına ilişkin veriler işletmenin e-ticaret ekibinin kayıt altına aldığı özel bir veri tabanından çekilmiştir. Öncelikle veri ön işleme adımları tamamlanmış ve bu kapsamda özelleştirilmiş fonksiyonlar yazılmıştır. Ürünlerin ilk adımda Metin Benzerliği algoritmasına dahil olarak uygun ürün önerileri bulunmuştur. Sonraki adımda TDA algoritmasına dahil olarak ürün öneri tablosu tamamlanmıştır ve elde edilen çıktılar csv formatında alınarak Google Tag Manager kodlarıyla sisteme eklenmiştir. Google Analytics üzerinden sonuçlar gözlemlenmiştir. Ayrıca Python kullanılarak Excel çıktısı üzerinden satış miktarları baz alınarak, her bir ürün için önerilen 3 ürünün mevcut duruma kıyasla değişim yüzdeleri hesaplanarak geliştirilen ürün öneri sisteminin etkileri hesaplanmıştır. Şekil 1'de çalışma adımlarının akışını gösteren şema verilmiştir.



Şekil 1. Uygulama Akış Şeması

2.1. Öneri Sistemleri

Ürün veya içerik öneri sistemlerindeki temel hedef, kullanıcıların beğenisini öngörüp onlara bu beğenileri neticesinde öneriler sunmaktır. Öneri sistemlerinde öneri sunulacak alana, elde edilen veri yapısına ve hedefine göre tercih edilen metotlar birkaç ana temel üzerinde ilerlemektedir. Bu metotların ve algoritmaların türleri ve birbirleri ile olan ilişkilerini gösteren, öneri sistemleri yapısı geçmiş çalışmalardan derlenerek akış diyagramı halinde Şekil 2'de verilmiştir.



Şekil 2. Öneri Sistemi Yöntemleri

Yapılan çalışmada işbirliğine dayalı filtreleme yöntemleri içerisinde bulunan kullanıcı tabanlı filtreleme yöntemi Tekil Değer Ayrışımı(TDA) kullanılarak ürün öneri tablosunun doğrudan etkileyen korelasyon değerleri hesaplanmıştır. Literatür incelemesi yapıldığında ürün tabanlı öneri sistemlerinde özellikle TDA'nın en iyi performans değerlerine sahip olduğu açıkça gözlemlenmiştir. TDA'nın geleneksel işbirlikçi algoritmalarından daha iyi sonuçlar verdiği, veri matrisindeki kısıt seyreklik problemi, kümeleme algoritmaları için benzerliklerin hesaplanmasında ve uygun komşulukların belirlenmesindeki sorunların giderilmesinde uygulanan algoritmalar arasında TDA'nın en iyi sonuçları verdiği tespit edilmiştir [15].

2.2. Metin Benzerliği Algoritması

Metin benzerliği algoritmaları yapılan işlemlerin niteliklerine göre sınıflandırılmaktadır. Belirteç Bazlı Benzerlik Sınıfı'nda beklenen girdi, tam dizelerden ziyade bir dizi ayıraçtır. Amaç mesafe ölçülecek her 2 sette de benzer ayıraçları bulmaktır. Ortak belirteç sayısı arttıkça, setler arasındaki benzerlik de artar. Kosinüs Benzerliği, Monge Elkan Mesafesi ve Bag Mesafesi, Jaccard Benzerlik Katsayısı ve Sorensen Dice Benzerlik Katsayısı ve Tversky İndeksi bu kategori içerisinde yer almaktadır. Diziliş Tabanlı Benzerlik Sınıfı'nda Ratcliff/Obershelp Benzerliği yer almaktadır. İki dize arasındaki ortak alt dizelerin bir faktörüdür. Amaç her 2 dizede bulunan en uzun diziyi bulmaktır. Bu dizelerden fazla bulunması durumunda benzerlik puanı daha yüksek olmaktadır. Uzaklık Düzenleme Bazlı Benzerlik Sınıfı'nda ise Levenshtein, Jaro-Winkler, Smith Waterman, Hamming, ve Needleman Wunsch Mesafeleri bulunmaktadır. Amaç bir dizeyi diğer dizeye dönüştürmek için değiştirme, ekleme, silme vb. gerekli işlem sayısının hesaplanmasıdır. Yapılan işlem sayısı arttıkça iki dize arasındaki benzerlik daha az gerçekleşmektedir [16].

Yapılan makale çalışmasında metin benzerliği algoritmaları içerisinde bulunan uzaklık düzenleme bazlı benzerlik sınıfına ait Levenshtein Mesafesi kullanılmıştır. Bu algoritmanın kullanılmasındaki amaç kullanılacak veride çok sayıda yanlış girilmiş ve düzenlenmesi gereken dizelerinin bulunması, ürün ismi içerisinde yer alan, istenilen nitelik olan model ailelerini ön plana çıkarabilmesi, uzman görüşleri sonrası bazı özellikler için ceza puanı uygulayarak uygun ürün önerilerin gösteriminde yüksek fayda sağlamasıdır.

2.2.1. Levenshtein Mesafesi

Levenshtein mesafesi iki dize arasındaki benzerliğin bir ölçüsüdür. Algoritma, a dizesini b dizesine dönüştürmek için gerekli minimum değişiklik sayısı olarak tanımlanır. Bu, a dizesine bir karakter ekleyerek, silerek veya değiştirilerek yapılır. Levenshtein mesafesi ne kadar küçük ise, dizeler de bir o kadar benzemektedir [17]. Formülasyonu aşağıda gösterildiği gibidir:

$$lev(a,b) = \begin{cases} max(i,j)\, lev_{a,b}\ (i,j) & , eğer\ min(i,j) = 0 \\ \begin{matrix} lev_{a,b}\ (i-1,j)+1 \\ lev_{a,b}\ (i,j-1)+1 \end{matrix} & , diğer \\ lev_{a,b}\ (i-1,j-1)+1_{(ai \neq bj)} & \end{cases} \quad (1)$$

Levenshtein mesafesinin hesaplanması için, her kelimeden geçmeli ve en az + 1 kullanılmalıdır.
(i-1, j): sol kutu anlamına gelir (silme).
(i, j-1): üst kutu anlamına gelir (ekleme).
(i-1, j-1): sol üst kutu anlamına gelir (yer değiştirme).

2.3. Tekil Değer Ayrışımı Algoritması

Tekil Değer Ayrıştırma (TDA), karmaşık bir matrisi daha basit matrislerin çarpımı şeklinde gösterilmesini sağlayan bir işlemdir. Bu işlemlerde en çok kullanılan algoritmalar, TDA, QR Ayrışımı, Alt-Üst Ayrıştırma, CS Ayrışımı, QLP Ayrışımı, Pivot QR Ayrışımı ve Pivot QLP Ayrışımı'dır [18].

Özellikle matrisin köşegenleştirilmesi ile sıkı bir ilişkisi olan TDA, tüm matris ayrışım algoritmalarının başında gelerek, özellikle regresyon analizinde önemli bir yer tutmaktadır [19].

$X=\begin{bmatrix}3.01 & 0.01 & -2.99\\ 2.99 & -0.01 & -3.01\\ 2.00 & -4.00 & 2.00\end{bmatrix}$ olsun.

X matrisi aşağıdaki gibi çözümlenebilir [18]:

$$X=\begin{bmatrix}3 & 0 & -3\\ 3 & 0 & -3\\ 0 & 0 & 0\end{bmatrix}+\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 2 & -4 & 2\end{bmatrix}+\begin{bmatrix}0.01 & 0.01 & 0.01\\ -0.01 & -0.01 & -0.01\\ 0 & 0 & 0\end{bmatrix} \quad (2)$$

$$=6\begin{bmatrix}1/2 & 0 & -1/2\\ 1/2 & 0 & -1/2\\ 0 & 0 & 0\end{bmatrix}+2\sqrt{6}\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 1/\sqrt{6} & -2/\sqrt{6} & 1/\sqrt{6}\end{bmatrix}+\sqrt{6}/100\begin{bmatrix}1/\sqrt{6} & 1/\sqrt{6} & 1/\sqrt{6}\\ -1/\sqrt{6} & -1/\sqrt{6} & -1/\sqrt{6}\\ 0 & 0 & 0\end{bmatrix} \quad (3)$$

$$=6\begin{bmatrix}1/\sqrt{2}\\ 1/\sqrt{2}\\ 0\end{bmatrix}\begin{bmatrix}1/\sqrt{2} & 0 & -1/\sqrt{2}\end{bmatrix}+2\sqrt{6}\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}\begin{bmatrix}1/\sqrt{6} & -2/\sqrt{6} & 1/\sqrt{6}\end{bmatrix} \quad (4)$$

$$+\sqrt{6}/100\begin{bmatrix}1/\sqrt{2}\\ -1/\sqrt{2}\\ 0\end{bmatrix}\begin{bmatrix}1/\sqrt{3} & 1/\sqrt{3} & 1/\sqrt{3}\end{bmatrix} \quad (5)$$

$$=\Theta_1U_1V_1+\Theta_2U_2V_2+\Theta_3U_3V_3=\sum_{i=1}^{3}\Theta_i\,U_i\,V_i^T \quad (6)$$

$$=\begin{bmatrix}1/\sqrt{2} & 0 & 1/\sqrt{2}\\ 1/\sqrt{2} & 0 & -1/\sqrt{2}\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}6 & 0 & 0\\ 6 & 2\sqrt{6} & 0\\ 0 & 0 & \sqrt{6}/100\end{bmatrix}\begin{bmatrix}1/\sqrt{2} & 0 & -1/\sqrt{2}\\ 1/\sqrt{6} & -2/\sqrt{6} & 1/\sqrt{6}\\ 1/\sqrt{3} & 1/\sqrt{3} & 1/\sqrt{3}\end{bmatrix} \quad =U\,\Theta V^T \quad (7)$$

Yukarıdaki denklem eşitliğinden görülmektedir ki, denklem (6)'daki U_i denklem (7)'deki U ortonormal matrisinin sütunudur. Denklem (6)'deki V^T ise, (7)'deki V^T ortonormal matrisinin satırıdır. Θ_i ise, Θ köşegen matrisinin köşegenindeki öğesidir. Denklem (6) ve (7) ise, X matrisinin tekil değer ayrışımının (TDA) açıklanmasıdır. Sadece buradaki X değil, her türlü matris denklem (7) gibi çözümlenebilmektedir.

2.4. K-Ortalamalar Algoritması

K-Ortalamalar algoritması, devam eden her aşamada kümelerin yenilendiği ve ulaşabileceği en optimum sonuç değerlerine kadar süregelen bir algoritma çeşididir. Bu alanda etkili ve ilk tercih edilen algoritmalardan biridir. Bölümlemeli diğer algoritmalar K-Ortalamalar algoritmasına benzemektedir veya bu algoritmadan esinlenmiştir. K-Ortalamalar algoritmasındaki k, önceden belirlenmiş olan ve aranan küme adedini ifade eder. k adet küme merkezi keyfi olarak algoritmanın başlangıç seviyesinde seçilir. Seçilen bu noktalar prototip olmak üzere isimlendirilmektedir. k adet prototip ve her bir küme olacak şekilde kümeleme analizinin başlangıcındaki durum 8 numaralı eşitlikte gösterildiği gibidir.

$$W_i =, \in \{1, \ldots, k\}, l \in \{1, \ldots, m\} \quad (8)$$

Algoritma girdileri, aşamaları ve elde edilen çıktı aşağıda gösterildiği gibi açıklanabilir:

Girdiler:

k: belirlenen küme adedi

D: mevcut verilere dair adımlar:

Keyfi olarak belirlenen k adet elemanın küme merkezi (s_1, s_2, …, s_k) olarak belirlenmesi
Her bir elamanın en yakın olduğu s_i'nin kümesine atanması
Kümelere ait s_1, s_2, …, s_k değerlerinin yeniden hesaplanması
Kümede farklılık bulunmayan dek birinci aşamadan itibaren ilerlenmesi.

Adımların tamamlanmasından sonra değişiklik bulunmuyorsa durulması gerekmektedir. Çıktılar: m adet küme biçiminde olacaktır [20]. Yapılan çalışmada fiyatlarına göre "Good", "Better" ve "Best" olmak üzere 3 adet küme oluşturulmuştur.

3 UYGULAMA

3.1. Verilerin elde edilmesi

Oluşturulması hedeflenen ürün öneri sistemi gerçek veriler üzerinden oluşturulmuştur. Kullanılacak algoritmaların doğru ve etkin bir şekilde çalışabilmesi ve önerilecek ürünlerin doğrudan müşteriye gösteriminin yapılabilmesi için detaylı ve kapsamlı bir veri ön işleme çalışması yapılmıştır.

Ürün grupları için algoritmanın doğru çalışması amacıyla bazı düzenlemeler yapılmıştır. Öneri sisteminin çalışmasından önce Python üzerinden ürünler ve ait oldukları ürün grupları incelenmiştir. Bazı ürünlerin boş ve hatalı geldiği tespit edilmiştir. Bu ürünlerin bağlı olduğu ürün grupları filtrelenerek analiz kapsamından çıkarılmıştır.
Tavsiyelerin daha doğru çalışması için ihtiyaç görülen durumlarda yeni ürün grupları oluşturulmuş ve ilgili ürünler bu gruplara atanmıştır.

E-ticaret ekibinden alınan veri üzerinde, satışa açık (aktif) ve stokta bulunma durumları incelenmiştir. Filtreleme işlemi yapılarak analiz çalışmasına dahil edilmiştir. Stoğu olmayan veya stokta bulunmasına rağmen e-ticaret satışına kapalı ürünlerin olması ihtimaline karşı bu ürünlerin öneri olarak müşterilere gösterimini engellemek için bu işlem yapılmıştır.

3.2. Metin benzerliği algoritmasının uygulanması

Bu algoritmanın kullanılmasındaki amaç ürün isimleri içerisinde geçen model ailelerinin tespit ederek birbirlerine olan uzaklıklarını tespit etmek ve benzer ürün önerilerini müşterilere sunmaktır.

Öncelikli olarak ürün isimleri incelenerek içerisinde geçen kelimelerden liste oluşturulmaktadır. Listenin oluşturulmasındaki temel amaç ise bu listede bulunan kelimeler ürün adı içerisinde bulunuyorsa hariç tutularak ürün isminden sadece model ailesinin getirilmesi ve yanlış ürün önerilerinin önüne geçilmesini sağlamaktır. Burada yer alan kelimeler bazı benzer olmayan ürünlerin birbirleri olan benzerlik uzaklığını azaltmakta ve ceza puanını düşürerek algoritmayı yanıltmaktadır. Bunun sonucunda da aynı ürün ailesinde bulunmamasına rağmen geliştirilen sistemin ürün önerilerini hatalı bir şekilde getirmesine neden olacaktır.

Algoritmanın çalışması sıralamaya bakılmaksızın kelimelerin benzerliğine göre ilerlemektedir. Karşılıklı 2 metne bakılarak skor puanları verilmektedir. En düşük skor puanına sahip ürün seçilerek ilerlenmektedir. Skor puanı ne kadar yüksekse o kadar cezalandırmış kabul edilmektedir. Ceza puanı filtresi 0,86 olarak belirlenmiştir.

Çok az sayıda üründe, oluşturulan listedeki kelimeler hariç tutulduğunda ürün alanında boş gelen değerler tespit edilmiştir. Programın hata vermesini engellemek amacıyla boş gelen değerler çıkarılmıştır. Şekil 3'de Metin Benzerliği algoritmasına ait yazılan kodların ekran görüntüsü paylaşılmıştır. Ürün öneri sisteminde tablonun öncelikli olarak doldurulması burada gerçekleşmiştir.

```
for i in product_recommendation_table["Product_Name"]:
    #ürün grup name alınması
    group_name=group_product_names[group_product_names["ProductDim[InventName]"]==i]["ProductDim[ProductGroup]"].values[0]
    #ürün segment değeri
    #segment_level=group_product_names[group_product_names["ProductDim[InventName]"]==i]["segment"].values[0]
    #segment_filtered_list=segment_filtreleme(Corr_List, segment_level)
    boyut=updated_reco_table[updated_reco_table.index== group_name].dropna(axis=1).shape[1]

    abc=' '.join([w for w in i.split(" ")[:9] if w not in liste and not w.isdigit()])
    print(abc)

    if (updated_reco_table[updated_reco_table.index== group_name].shape[0])!=0:

        for k in np.arange(boyut):
            reco_group=updated_reco_table[updated_reco_table.index== group_name].dropna(axis=1).iloc[0,k]
            first_product=[]
            second_product=[]
            distance_score=[]


            for j in df_karaca[df_karaca["ProductDim[ProductGroup]"]==reco_group]["ProductDim[InventName]"].unique():
                #print(k)

                if (' '.join([w for w in i.split(" ")[:9] if w not in liste and not w.isdigit()])=='') or \
    (' '.join([w for w in j.split(" ")[:9] if w not in liste and not w.isdigit()])==''):
                    i
                    #print("++++++++++++",i,"-----------",j)
                else:

                    distance=documentSimilarity(' '.join([w for w in i.split(" ")[:9] if w not in liste and not w.isdigit()]), '
                    first_product.append(i)
                    second_product.append(j)
                    distance_score.append(distance)


            similarity_distance=pd.DataFrame({"first_product":first_product, "second_product":second_product,
                                              "distance_score":distance_score})

            if similarity_distance.sort_values("distance_score").shape[0]!=0:

                #print(similarity_distance[similarity_distance["first_product"]==i].sort_values("distance_score").values[0].resh
                first,second, distance=similarity_distance.sort_values("distance_score").values[0]

                if distance<0.86:
                    #print(first,"---",second,"---",distance)
                    index_value=product_recommendation_table[product_recommendation_table["Product_Name"]==i].index.values[0]
                    product_recommendation_table.iloc[index_value,k+1]=second+"..."
    else:
        print(group_name)
```

Şekil 3. Metin Benzerliği Algoritmasının Uygulanması

3.3. TDA(Tekil Değer Ayrışımı) algoritmasının uygulanması

Metin Benzerliği algoritmasının çalışmasından sonra ürün öneri tablosundaki çoğu alan boş olarak görülmektedir. Bu alanların doldurulması için öncelikli olarak tüm ürünlerin TDA algoritmasına girmesi sağlanmaktadır.

TDA(Tekil değer ayrıştırması) algoritması müşteri ID'lerini ve ürün adlarını temel alarak ilerlemektedir. Kullanıcı tabanlı olmayıp ürün içerik tabanlı ilerlemektedir. Müşteri ID'lerinin kullanılmasındaki amaç birlikte alınan ürünleri yakalamaktır ama hangi ürünün kim tarafından alındığı çalışmanın kapsamında bulunmamaktadır.

Mağazadan müşteri kaydı olmadan yapılan alışverişlerde veya benzeri durumlarda müşteriye bağlı satış bulunamadığı için boş gelen değerler bulunmaktadır. Algoritmanın çalışmasını olumsuz yönde etkilememesi için fatura numaraları bulunan ama müşteri ID'si bulunmayan ilgili satırlar filtrelenmektedir.

Şekil 4'de TDA algoritmasının uygulanmasına yönelik yazılan Python kodlarının bazı kısımları ekran görüntüsü olarak paylaşılmıştır.

```
#tümü 1'Den oluşan yeni sütun ekle
item_user["rating"]=1
```

```
item_user.rename(columns={"ContactDim[ContactId]":"user_id", "ProductDim[InventName]":"item_id"}, inplace=True)
```

```
#item_user = item_user.rename(columns={"ContactDim[ContactId]":"user_id", "ProductDim[InventName]":"item_id"})
```

```
#tek 1 defa alışveriş yapan customer
user_list=pd.DataFrame(item_user["user_id"].value_counts())
user_list[user_list["user_id"]==1]
```

```
# tek bulunan customer ID'lerin çıkarılması

item_user=item_user[item_user['user_id'].duplicated(keep=False)].reset_index(drop=True)
item_user
```

```
alışveriş_sayıları=item_user.groupby("user_id").agg("count").reset_index()
```

```
user_id_liste=alışveriş_sayıları[(alışveriş_sayıları["item_id"]<80) & (alışveriş_sayıları["item_id"]>2)]["user_id"].values
```

```
user_id_liste
```

```
item_user=item_user[item_user["user_id"].isin(user_id_liste)].reset_index(drop=True)
```

```
item_user
```

```
# user_item pivot table oluşturulması

rating_crosstab = item_user.pivot_table(values='rating', index='user_id', columns='item_id', fill_value=0)
#rating_crosstab.head()
```

```
#sadece kontrol boyutlar için
rating_crosstab.shape
```

```
#transpozunu aldık
X = rating_crosstab.T
```

```
SVD = TruncatedSVD(n_components=40, n_iter=7, random_state=5)
resultant_matrix = SVD.fit_transform(X)
resultant_matrix.shape
```

```
resultant_matrix[0]
```

```
### correlation matrix
corr_mat = np.corrcoef(resultant_matrix)
corr_mat.shape
```

Şekil 4. TDA Algoritmasının Uygulanması

Alışveriş yaptığı ürünleri, müşterilerin beğendiği kabulü ile ilerlenmektedir ve tüm müşteri ID'lerine aldığı her bir ürün için matriste 1 değeri atanmaktadır.

Tek bir ürün alan müşteriler ürün önerisine fayda sağlamayacağı ve işlemsel olarak yük getireceği için çıkartılmaktadır. Ayrıca çoğunluğu personellere ait olduğu tespit edilen, müşteri ID üzerinde 80 adetten fazla ürün bulunan satışlar ve bir müşteriye ait 3'den az alışveriş olduğu durumlar da çıkartılmaktadır.

Müşterilerin aldığı ürünlerin bulunduğu sütuna "1", diğer sütunlara "0" değeri verilmektedir. Örneğin; Bir müşteri 5 ürün almış ise ilgili satır bazında ilişkili olduğu 5 sütuna "1" değeri verilirken diğer tüm sütunlara "0" değeri atanmaktadır. Bu şekilde ilerlenerek yaklaşık 300 bin satır(müşteri ID) ve 2800 sütunluk(ürün) bir matris oluşturulmaktadır.

Matrisin transpozu alınarak 40 bileşenli ve 7 iterasyonlu bir işlem uygulanmaktadır.

Ürünler için de ürün sayısı kadar satır ve sütun oluşturarak karşılıklı geldiği değere 1 ve diğer alanlara 0-1 arası değerler getirilmektedir. Burada da yaklaşık [2800,2800] boyutunda bir matris oluşturulmaktadır. Ürün için matris çarpımı yaparak korelasyon değerleri elde edilmektedir. Bu sayede ürünlerin birbirleri ile ne kadar ilintili olduğunun tespiti yapılmaktadır.

Kullanıcı sayısının çok büyük olduğu durumlarda İşbirlikci Filtreleme(kullanıcıya bağlı) yönteminin kullanılması zorlaşmaktadır. Bu problemi aşmak için öneri sistemlerinde kullanıcılar değil ürünler arasındaki ilişkiden yararlanılması amaçlanmıştır.

3.4. Segment ve fonksiyonların oluşturulması

Bu aşamada en doğru ürün önerilerinin yapılabilmesi için ürün isimleri üzerinde gidilerek renk ve marka fonksiyonları yazılmıştır, ürünler fiyat bilgisine göre segmentlere ayrılmıştır.

Ürün isimlerinde geçen renklere dair bilgiler incelenerek fonksiyon tanımlanmıştır. Buradaki amaç bir renge ait kısaltma, İngilizce karakter veya yanlış yazımdan kaynaklı oluşan farklılıkları gidererek renk bilgisinin doğru bir şekilde algoritmada kullanılmasını sağlamaktır. Bir örnek üzerinden incelenecek olursa; "Su Yeşili" rengi ürün isimlerinde "S.YESIL", "SU YESILI", "AQUA GREEN" gibi farklı olarak yazılabilmektedir. Burada ürün isimlerinde geçen farklılıkları gidererek "SU YEŞİL" olarak tanımlamasını yapılmıştır. Renk bilgisi yazılmayan ürünlerde ise "RENKSİZ" tanımlaması yapılmıştır.

Ürün isimlerinde geçen marka adlarının farklı olarak geldiği gözlemlenmiştir. Aynı marka ürün önerilerinin yapılabilmesi için yeni bir sütun oluşturularak ürünlere ait marka tanımlamaları yapılmıştır. Bir örnek üzerinden incelenecek olursa; ürün isimleri içerisinde geçen "KAID", "Kitchenaid", "KitchenAid" olarak farklı gelmesinin önüne geçilerek bu ürünlerin tümü için "KITCHENAID" marka tanımlaması yapılmıştır.

Ürünlerin ortalama satış fiyatları belirlendikten sonra K-Ortalamalar algoritması kullanılarak "Good", "Better" ve "Best" segmentleri oluşturulmuştur. Bu işlem yapılarak aynı segmentten ürün önerileri kullanıcılara sunulması amaçlanmıştır. Ayrıca firmanın değer gördüğü bazı özel ürün grupları için doğrudan segment tanımlamaları yapılarak fiyat segmentinden dolayı öneri sisteminin düşük fiyatlı ürün önerilerini sunmasının önüne geçilmiştir.

```
from sklearn import cluster
```

```
def kmeans_segment(segment_key):
    if segment_key==good_segment:
        segment="Good"
    elif segment_key==best_segment:
        segment="Best"
    else :
        segment="Better"
    return segment
```

```
# 3'den az çeşiti olan grupların ürünlerin kaldırılması

aaa=df_karaca.groupby("ProductDim[ProductGroup]")["ProductDim[InventName]"].nunique()
aaa=aaa.reset_index()
fazla_çeşit_olan_gruplar=aaa[aaa["ProductDim[InventName]"]>=3]["ProductDim[ProductGroup]"].values
aaa=aaa[aaa["ProductDim[InventName]"]<3]["ProductDim[ProductGroup]"]
az_çeşitli_gruplar=aaa.values
df_karaca=df_karaca[~df_karaca["ProductDim[ProductGroup]"].isin(aaa)].reset_index(drop=True)
```

```
group_product_names["segment"]=""
```

```
for i in fazla_çeşit_olan_gruplar:
    prices=group_product_names[group_product_names["ProductDim[ProductGroup]"]==i]["Price"].reset_index(drop=True)
    k_means = cluster.KMeans(n_clusters=3)
    k_means.fit(np.array([prices]).reshape(-1, 1)) #K-means training
    y_pred = k_means.predict(np.array([prices]).reshape(-1, 1))
    pred = pd.DataFrame(y_pred)
    pred.columns = ['segment_key']
    # we merge this dataframe with df
    prediction = pd.concat([prices,pred], axis = 1)
    good_segment=prediction[prediction["Price"]==prediction["Price"].min()]["segment_key"].values[0]
    best_segment=prediction[prediction["Price"]==prediction["Price"].max()]["segment_key"].values[0]
    prediction["segment"]=prediction["segment_key"].apply(kmeans_segment)
    prediction=prediction.drop(columns=("segment_key"))
    group_product_names.loc[group_product_names["ProductDim[ProductGroup]"]==i,"segment"]=prediction["segment"].values
```

```
group_product_names.loc[group_product_names["segment"]=="","segment"]="Better"
```

Şekil 5. K-Ortalamalar Algoritması ile Fiyat Segmentasyonunun Yapılması

3.5. TDA algoritması ve oluşturulan segmentlere göre tavsiye ürünlerin belirlenmesi

Metin Benzerliği algoritmasının çalışmasından sonra ürün öneri tablosunda boş kalan alanlar için TDA algoritması sonucu elde edilen korelasyon değerleri ve oluşturulan segmentler değerlendirmeye alınarak ürün önerileri tamamlanmaktadır. Şekil 6'da ürün önerilerinin belirlenmesine yönelik yazılan Python kodlarının bir kısmı ekran görüntüsü olarak paylaşılmıştır.

```
for i in product_recommendation_table["Product_Name"]:
    #print(i)

    if i in (rating_crosstab.columns):
        col_idx = rating_crosstab.columns.get_loc(i)
        corr_specific = corr_mat[col_idx]
        Corr_List=pd.DataFrame({'corr_specific':corr_specific, 'items': rating_crosstab.columns})\
            .sort_values('corr_specific', ascending=False)
        Corr_List=pd.merge(Corr_List, group_product_names, left_on="items", right_on="ProductDim[InventName]"
                           ).drop(columns="ProductDim[InventName]")
        group_name=group_product_names[group_product_names["ProductDim[InventName]"]==i]["ProductDim[ProductGroup]"].values[0]
        segment_level=group_product_names[group_product_names["ProductDim[InventName]"]==i]["segment"].values[0]
        segment_filtered_list=segment_filtreleme(Corr_List, segment_level)
        color_level=group_product_names[group_product_names["ProductDim[InventName]"]==i]["color"].values[0]
        segment_filtered_list=color_filtreleme(segment_filtered_list, color_level)
        boyut=updated_reco_table[updated_reco_table.index== group_name].dropna(axis=1).shape[1]


        for k in np.arange(boyut):
            reco_group=updated_reco_table[updated_reco_table.index== group_name].dropna(axis=1).iloc[0,k]
            #print(i)
            #print(reco_group)

            # Bazı KEA ürünleirin marka uyumu
            if group_name in ["TOST MAKİNESİ","SMOOTHIE BLENDER","MUTFAK ŞEFİ","MUTFAK ROBOTU","EL BLENDERI","DOĞRAYICI","EL MİKS
                marka_level=group_product_names[group_product_names["ProductDim[InventName]"]==i]["marka"].values[0]
                #print(i)
                #print(group_name,"----------------------",marka_level)
                segment_filtered_list=marka_filtreleme(segment_filtered_list, marka_level)

            if segment_filtered_list[segment_filtered_list["ProductDim[ProductGroup]"]==reco_group].shape[0]==0:
                print(reco_group,"ürün gelmedi")
            else:
                product=segment_filtered_list[segment_filtered_list["ProductDim[ProductGroup]"]==reco_group]["items"].values[0]
                index_value=product_recommendation_table[product_recommendation_table["Product_Name"]==i].index[0]
```

Şekil 6. TDA ve segmentlere göre ürün önerileri alanının doldurulması

Tamamlayıcı ürünler tavsiye edilirken tekli ürünlere aynı model farklı boy veya aynı model farklı çeşit ürünlerin getirilmesi sağlanmıştır. Şekil 7'de gösterildiği gibi "Tava" ürün grubu için yine aynı ürün grubundan öneriler yapılabilmesi sağlanmaktadır. Burada aynı ölçülerde ürün önerilmemesi için farklı boyutlarda gelmesi kriteri de eklenmiştir. Bu ve benzeri birçok ürün grubu için de özel tanımlamalar yapılmıştır.

```
if (group_name=="TAVA") and (reco_group=="TAVA"):
    first_product=[]
    second_product=[]
    distance_score=[]

    for j in group_product_names[group_product_names["ProductDim[ProductGroup]"]==reco_group]["ProductDim[InventName
        a="".join([str(x) for x in i if x.isdigit()])
        b="".join([str(x) for x in j if x.isdigit()])

        if (a=="") or (b==""):
            distance=documentSimilarity(i,j)
            if distance>0.6:
                first_product.append(i)
                second_product.append(j)
                distance_score.append(distance)

        else:
            number_check_distance=documentSimilarity("".join([str(x) for x in i if x.isdigit()]),"".join([str(x) for

            if (number_check_distance>0) and (a not in b) and (b not in a):
                distance=documentSimilarity(i,j)
                first_product.append(i)
                second_product.append(j)
                distance_score.append(distance)

    similarity_distance=pd.DataFrame({"first_product":first_product, "second_product":second_product,
                                      "distance_score":distance_score})
    product=similarity_distance[similarity_distance["distance_score"]>0].sort_values("distance_score").reset_index(d

    if k==0:
        product=similarity_distance[similarity_distance["distance_score"]>0].sort_values("distance_score").reset_ind
    elif k==1:
        product=similarity_distance[similarity_distance["distance_score"]>0].sort_values("distance_score").reset_ind

    product_recommendation_table.iloc[index_value,k+1]=product
```

Şekil 7. Tava Ürün Grubuna Yönelik Düzenleme Örneği

3.6. Program çıktılarının elde edilmesi, web sitesi ve mobil uygulamada ilgili alanlara aktarılması

Ürün öneri sistemine ait tablonun tamamlanmasının ardından her bir ürün için 3 tamamlayıcı ürün önerisi getirilmiştir. Bu ürünler site içerisine aktarılmadan önce ürün isimlerine karşılık gelen ürün ID'lerine göre çevrilmektedir.

Sonraki adımda elde edilen listenin csv formatında çıktısı alınarak web sitesi ve mobil uygulamada ilgili alanı besleyecek şekilde eklenmektedir.

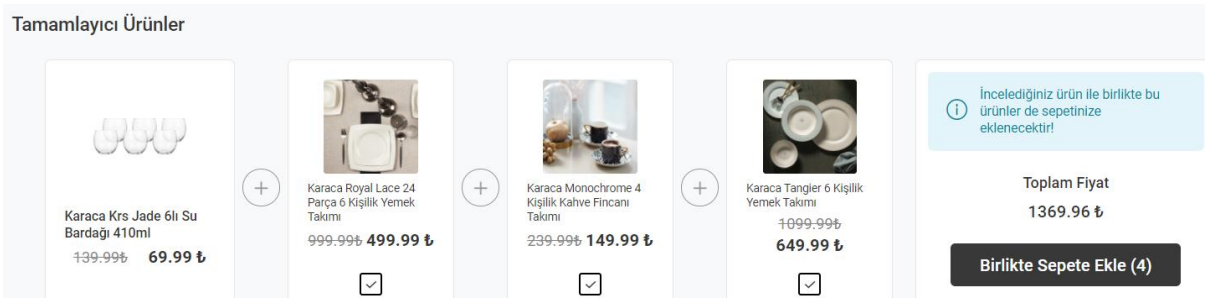
Google Analytics üzerinden, kullanıcıların geliştirilen ürün öneri sisteminin etkisini mevcut sisteme kıyasla değişimini ölçümlemek amacıyla sitede ve mobil uygulamada tamamlayıcı ürünler alanının görüntülenme ve tıklanma sayılarının karşılaştırılması yapılmak istenmektedir. Ürün sayfasında yer alan bu bilgiler Google Tag Manager üzerinden "Event" tanımlamaları yapılarak Google Analytics'e bağlanmaktadır. GTM ile köprü kurularak Google Analytics'e sürekli bir veri akışı sağlanmaktadır. İlgili alanlarda görüntülenme ve tıklanma sayılarını almak için Tag'ler oluşturulmuştur. Sayfada ilgili alan %51 ve üzeri oranında görüntülendiği zaman GTM üzerinden event tetiklenecek şekilde Google Analytics'e yazılmaktadır. Yeni ürün öneri sistemi uygulamaya alınmadan önce ilgili alanların tanımlamaları yapılarak yeni sistemin performansını gözlemlemek amaçlanmıştır.

4 BULGULAR

Ürün öneri sisteminin geliştirilme ihtiyacı ilk olarak mevcut durumda bulunan ürün önerilerinin eksik ve ilgili olmayan ürünlerin gösteriminin tespiti ile ortaya çıkmıştır. Bu kısımda, geliştirilen ürün öneri sisteminin kullanıcı ekranlarının mevcut duruma göre kıyaslanması, tıklanma ve görüntülenme sayılarındaki değişim ve tamamlayıcı olarak satışlarının yüzdesel farklılıkları analiz edilerek sistemin performansı değerlendirilecektir.

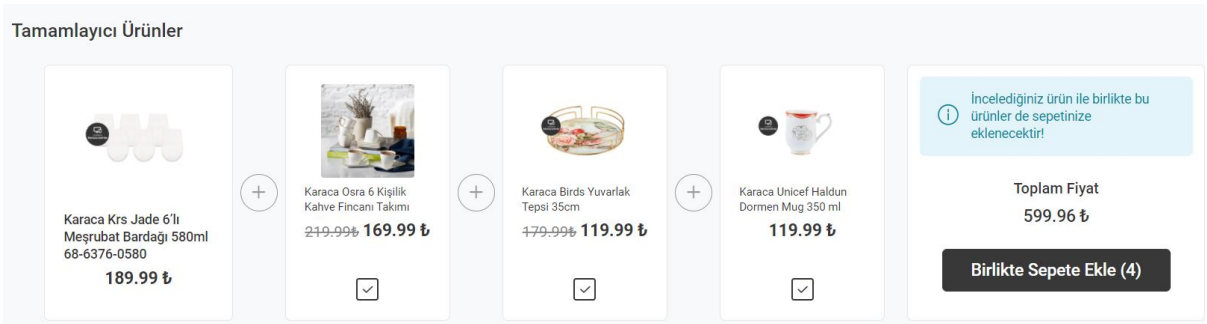
4.1. Kullanıcı ekranlarının karşılaştırılması

Mevcut durumda önerilen ürünlerin, ürün öneri sisteminde kullanıcılara tavsiye olarak gösteriminde eksik ve mantıksız ürün önerileri gösterdiği tespit edilmiştir. Şekil 8'de gösterildiği gibi düşük fiyat segmentine sahip su bardağı seçmiş bir müşteriye mevcut sistem, yüksek fiyatlı yemek takımı önerebilmekte ve bunun sonucunda da istenen satışların gerçekleşmemesi ve müşteri deneyiminin olumsuz yönde etkilenmesi gerçekleşmektedir. Ayrıca mevcut sistemin bazı ürünlerde de eksik ürün önerileri yaptığı tespit edilmiştir.



Şekil 8. Mevcut sistemin ürün önerileri

Yeni geliştirilen öneri sistemi mobil uygulama ve e-ticaret sitesinde uygulamaya alındıktan sonra tamamlayıcı ürünleri birbirine benzer, aynı fiyat segmentinde, istenilen ürün gruplarında ve sepete eklenen bir ürüne 3 ürün önerisini eksiksiz olarak getirdiği gözlemlenmiştir. Şekil 9'da gösterildiği gibi kullanıcıya ilgili ürünlerin birbirleri ile olan ilişkileri, fiyat vb. özellikler dikkate alınarak önerilmiş olduğu gözlemlenmektedir. Ayrıca farklı örnekler incelendiğinde 3 tamamlayıcı ürünün de eksiksiz olarak ilgili alana getirildiği belirlenmiştir.



Şekil 9. Geliştirilen sistemin ürün önerileri

4.2. Öneri sistemi performansının Google Analytics aracılığıyla karşılaştırılması

Google Analytics sonuçları incelendiğinde sistemin uygulanmaya başlandığı tarih olan 3 Ocak 2022'den 2 Şubat 2022 tarihine kadar olan görüntülenme ve tıklanma sayıları bir önceki aya göre gün bazında kıyaslanmıştır. Performans karşılaştırılması yapılacak olan bu tarihlerin, firmanın özel olarak tanımladığı satışların yüksek olarak gerçekleştiği bazı özel dönemler (Black Friday, Anneler Günü vb.) haricinde olmasına da dikkat edilmiştir. Bu sayede elde edilen sonuçların öneri sisteminin performansına yönelik yanıltıcı etki göstermesinin önüne geçilmiştir.

Şekil 10'de görüleceği üzere 1 önceki ay ile kıyaslandığında ürün öneri sistemi alanının görüntülenme sayılarında %7,17 artış gözlemlenirken, tıklanma sayıları incelendiğinde %28,93 olarak artış gerçekleşmiştir. Bu sonuçlar neticesinde ürünlerin tıklama sayılarının görüntülenme sayılarına oranla daha yüksek çıkması, ilgili alanda yapılan ürün önerilerinin kullanıcılar için daha uygun olduğunun ve yapılan çalışmanın olumlu yönde katkıda bulunduğunun göstergesi olarak kabul edilmektedir.

	Etkinlik İşlemi	Toplam Etkinlik Sayısı
		%6,75 ▲ 812.387 ve 761.042
☐	1. Viewed	
	03.Oca.2022 - 02.Şub.2022	**802.232** (%98,75)
	03.Ara.2021 - 02.Oca.2022	**748.561** (%98,36)
	Değişiklik %	**%7,17**
☐	2. Clicked	
	03.Oca.2022 - 02.Şub.2022	**16.139** (%1,97)
	03.Ara.2021 - 02.Oca.2022	**12.518** (%1,64)
	Değişiklik %	**%28,93**

Şekil 10. Google Analytics Görüntülenme ve Tıklanma Sayıları Karşılaştırması

3 Ocak 2022 tarihinden itibaren sitede ilgili alanda "Clicked" sayılarında günlük bazda bir önceki aya göre değişimleri kıyaslandığında Şekil 11'de görüleceği üzere mevcut sisteme göre her gün için daha iyi performans gösterdiği gözlemlenmiştir. Bir önceki sene ile kıyaslama, ilgili alana ait GTM kodları ile tanımlamaları o dönem olmadığı için yapılamamaktadır.

Şekil 11. Öneri Sisteminin Uygulandığı Tarihten(3 Ocak) İtibaren "Clicked" Sayılarının Bir Önceki Aya Göre Değişimi

4.3. Öneri sisteminin satışa olan etkisinin yüzdesel değişimi

Çalışmada geliştirilen sistemde bir ürüne ait 3 farklı ürün önerisi sunulmaktadır. Burada yeni sistemin performansı karşılaştırılırken, belirlenen ürüne yönelik elde edilen satış miktarları üzerinden tamamlayıcı ürünlerin beraber satılan ürün sayısı baz alınarak mevcut sisteme göre yüzdesel farkları hesaplanmıştır. Buradaki amaç yeni ürün öneri sistemi oluşturulduktan sonra tavsiye edilen tamamlayıcı ürünlerin performansını tespit etmektir.

Tüm set halinde satılan ürünlerde yeni sistemin mevcut sisteme göre artış oranı %7,62 olarak ölçülürken, tek olarak satılan ürünlerde artış oranı %11,2 olarak hesaplanmıştır. Tablo 1'de set halinde satılan ürünlere ait örnek veri grubu, Tablo 2'de ise tekli olarak satılan ürünlere ait örnek veri grubu paylaşılmıştır.

Tablo 1. Set halinde satılan ürünlere ait örnek veri grubu

Ürün Adı	Tamamlayıcı Ürünlerin Toplam Satış Oranları		Tamamlayıcı Ürün Satışları Artış Oranı
	Yeni Öneri Sistemi	Eski Öneri Sistemi	
Pure 6 kşlk ymt rnd	10,85%	2,62%	8,23%
Blendfıt 4 in 1 set red gold	9,18%	3,71%	5,47%
Stea 24 prç ymt	10,79%	1,66%	9,13%
Point tree 24 prç ymt	6,58%	1,34%	5,24%
Biogranit blackgold 7 prç tencere seti	12,71%	5,16%	7,55%
Overcut 6 prc bıçak seti	14,87%	2,81%	12,06%
Woodland herbal 6 parça servis seti	8,22%	1,31%	6,91%
Bio diamond master 3 prc tencere seti	12,09%	3,42%	8,67%
Karaca pop art 12 prç çay seti	14,62%	4,19%	10,43%
Kh rubi siyah cift kisilik ytk örtüsü set	13,09%	3,47%	9,62%

Tablo 2. Tekli olarak satılan ürünlere ait örnek veri grubu

Ürün Adı	Tamamlayıcı Ürünlerin Toplam Satış Oranları		Tamamlayıcı Ürün Satışları Artış Oranı
	Yeni Öneri Sistemi	Eski Öneri Sistemi	
Zeugma 25cm servis tabağı	15,48%	4,89%	10,59%
Mesopotamia bakır pilav tenceresi 18 cm	18,55%	3,71%	14,84%
Elvis 8 şef bıçağı	16,86%	3,37%	13,49%
Frida kahlo kase	9,75%	2,49%	7,26%
Lulya dekoratif tabak	17,85%	8,98%	8,87%
Mare pasta tabağı	16,90%	3,57%	13,33%
Topic tribal 280ml saklama kabı	9,63%	1,49%	8,14%
Berry dalgalı kek kalıbı 28 cm	14,89%	4,17%	10,72%
Black gold küçük sunum tabağı	15,32%	2,53%	12,79%
Karaca spin kesme tahtası-s	17,55%	2,64%	14,91%

Örnek veri gruplarında ve toplam veri setinde tek olarak satılan ürünler incelendiğinde geliştirilen ürün öneri sisteminin mevcut duruma kıyasla daha iyi sonuçlar verdiği gözlemlenmiştir. Burada tekli olarak satılan ürünlerin fiyat ortalamalarının daha düşük olması set halinde satılan ürünlere göre yüksek performans göstermesinin nedenlerinden biri olarak kabul edilmektedir.

5 SONUÇLAR

Öneri sistemleri son zamanlarda e-ticaret alanının hızla gelişmesiyle birlikte öne çıkan bir uygulama olarak görülmektedir. Uygulandığı firmanın dinamiklerine göre hazırlanması son derece önem arz etmektedir. Yapılan çalışmada ağırlıklı olarak züccaciye ve ev tekstili alanında satış yapan bir perakende firmasına yönelik ürün öneri sistemi geliştirilmiştir. Geliştirilen ürün öneri sisteminin mevcut duruma kıyasla web sitesi ve mobil uygulamada ilgili alanın görüntülenme ve tıklanma sayıları Google Analytics'den elde edilen veriler ile karşılaştırılmıştır. Yeni sistemin mevcut duruma kıyasla daha iyi performans gösterdiği tespit edilmiştir. Ayrıca ürünlere önerilen tamamlayıcı 3 tavsiye ürünün beraber satış adetleri üzerinden python aracılığıyla yüzdesel farkları hesaplanmıştır. Burada da tekli satılan ürünlerde %11,2 ve set halinde satılan ürünlerde %7,62 artış sağlanarak olumlu yönde katkı sağladığı gözlemlenmiştir.

Oluşturulan ürün öneri sisteminin etkilerinin daha iyi ölçümlenebilmesi için uzun süreli olarak incelenmesi gerekmektedir. Günümüzde etkisini fazlasıyla hissettiğimiz pandemi dönemi ve döviz kuruna bağlı olarak yaşanan fiyat değişiklikleri gibi durumlar satın alma davranışlarını etkilediği için farklı dönemlere ait karşılaştırmalar, oluşturulan öneri sisteminin performansının ölçülmesini ve geliştirmeye açık olan yönlerinin daha net bir şekilde ortaya çıkmasını sağlayacaktır.

Ayrıca gerçek ortamda uygulanabilir durumda olması sebebiyle sürekli gözlem yapma imkanı bulunmaktadır. Yapılacak olan değerlendirmeler sonucu ürün önerilerinin anlam bakımından hatalı bulunması veya uzman görüşlerinin olumsuz olması durumunda verilerde düzenleme, yeni ürün grupları oluşturma ve gerekli filtreleme işlemlerinin yapılmasına esneklik sağlayabilmektedir.

Gelecek çalışmalarda RFM analizi uygulanarak müşteri segmentlerinin oluşturulması ve ürün temelli önerilere destek olacak şekilde müşteri davranışlarının dahil edildiği hibrit bir ürün öneri sistemi geliştirilmesi amaçlanmaktadır. Hibrit sisteme uygun farklı algoritmalar kullanılarak mevcut çalışma ile performans karşılaştırılması yapılabilecektir.

Yazar Katkıları

Alper KİRAZ: Metodoloji, Yazılım, Doğrulama, Araştırma, Materyaller / Kaynaklar, Veri İyileştirme, Yazım - Özgün Taslak, Yazım - Değerlendirme & Düzenleme, Süpervizyon, Proje yönetimi
Bilal ERDEMİR: Kavramlaştırma, Yazılım, Veri analizi, Araştırma, Materyaller / Kaynaklar, Veri İyileştirme, Yazım - Özgün Taslak, Görselleştirme, Proje yönetimi, Finansman temini
Yazarlar makalenin son halini okuyup onaylamışlardır.

Çıkar Çatışması Beyanı
Yazarlar herhangi bir çıkar çatışması olmadığını beyan eder.

Kaynakça

[1] G. Adomavicius, A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6. 2005.

[2] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, "Item-based collaborative filtering recommendation algorithms", Proceedings of the tenth international conference on World Wide Web - WWW ’01, New York., 285-295, 2001.

[3] C. A. Gomez-Uribe, N. Hunt, “The netflix recommender system: Algorithms, business value, and innovation,” ACM Trans. Manag. Inf. Syst., vol. 6, no. 4, 2015.

[4] B. Smith, G. Linden, “Two Decades of Recommender Systems at Amazon.com,” IEEE Internet Comput., vol. 21, no. 3, 2017.

[5] S. Qin, R. Menezes, M. Silaghi, “A recommender system for youtube based on its network of reviewers”, 2010.

[6] M. Schedl, H. Zamani, C. W. Chen, Y. Deldjoo, M. Elahi, “Current challenges and visions in music recommender systems research,” Int. J. Multimed. Inf. Retr., vol. 7, no. 2, 2018.

[7] E. Yalçın, “İşbirlikçi Filtreleme Algoritmalarının Çok-Beğenilen Ürünlere Yönelik Yanlılığı”, BŞEÜ Fen Bilimleri Dergisi, vol. 8, no. 1, pp. 279-291, 2021.

[8] L. Zhaoyang, “Matris Ayrışımı,” İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü, Ekonometri Anabilim Dalı, Yüksek Lisans Tezi, 1993.

[9] A. Dundar, A. Gorgulu Kakisim, “Kıyafet Öneri Sistemi için Giyim Metaverilerine Dayalı Temsil Öğrenimi”, Avrupa Bilim ve Teknoloji Dergisi, vol. 29, pp. 105-110. 10, 2021.

[10] Y. K. Choi and S. K. Kim, “An auxiliary recommendation system for repetitively purchasing items in E-commerce,” 2014, doi: 10.1109/BIGCOMP.2014.6741415.

[11] H. Alharthi, D. Inkpen, and S. Szpakowicz, “A survey of book recommender systems,” J. Intell. Inf. Syst., vol. 51, no. 1, 2018.

[12] S. Guler, “Öneri Sistemleri ve E-Ticarette Öneri Sistemlerinin Kullanımı”, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar ve Bilişim Mühendisliği, Yüksek Lisans Tezi. 2019.

[13] M. Uçar, "E-Ticaret Siteleri için Bulanık Mantık ve Makine Öğrenmesi Tabanlı bir Öneri Sistemi", Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar ve Bilişim Mühendisliği, Yüksek Lisans Tezi, 2021.

[14] E. Yıldız, "Müşteri Segmentasyonu Odaklı Bütünleşik Bir Ürün Öneri Sistemi: Moda Perakendesi Sektöründe Bir Uygulama", Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Endüstri Mühendisliği, Yüksek Lisans Tezi, 2018.

[15] B. Batmaz, "Yükseköğretimde Öneri Sistemlerine Dayalı Ders Seçme Modeli", Eskişehir Osmangazi Üniversitesi, Fen Bilimleri Enstitüsü, İstatistik Bilgi Sistemleri, Doktora Tezi, 2018.

[16] Ü. Keleş, N. Duru, "Metin Benzerliği Algoritmaları ile Veri Tekilleştirme: Oteller Veri Tabanında Bir Uygulama", Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi, vol:5, no:1, 2021.

[17] R. T. Ionescu, M. Popescu, "Knowledge Transfer between Computer Vision and Text Mining", Switzerland: Springer, 2016.

[18] Z. Li, "Matris Ayrışımı", İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü. Ekonometri Anabilim Dalı, Yüksek Lisans Tezi, 2006.

[19] T. F. Abidin, B. Yusuf, M. Umran, "Singular Value Decomposition for Dimensionality Reduction in Unsupervised Text Learning Problems", 2nd International Conference on Education Technology and Computer (ICETC), vol. 4, pp. 422–426, 2010.

[20] J. Han, J. Pei, M. Kamber, "Data mining: concepts and techniques", 3rd Edition, Morgan Kaufmann Publishers, vol.7, no.6, 2011.