Pamukkale Univ Muh Bilim Derg, 25(5), 615-620, 2019

Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi

Pamukkale University Journal of Engineering Sciences

Yazılım geliştirme taleplerinin metin madenciliği yöntemleriyle önceliklendirilmesi

Prioritization of software development demands with text mining techniques

Murat Can TEKİN[1], *Volkan TUNALI*[2*]

[1]Bilgisayar Mühendisliği Bölümü, Fen Bilimleri Enstitüsü, Maltepe Üniversitesi, İstanbul, Türkiye.
muratcantekin@hotmail.com

[2]Yazılım Mühendisliği Bölümü, Mühendislik ve Doğa Bilimleri Fakültesi, Maltepe Üniversitesi, İstanbul, Türkiye.
volkan.tunali@gmail.com

Geliş Tarihi/Received: 19.11.2018, Kabul Tarihi/Accepted: 13.02.2019 doi: 10.5505/pajes.2019.47827
* Yazışılan yazar/Corresponding author Araştırma Makalesi/Research Article

Öz

Kurumsal şirketlerde, yazılımlardaki hatalar ve değişiklik talepleri genellikle bir talep yönetim sistemi üzerinden Bilgi Teknolojileri (BT) birimine iletilir. Bu sistemde yer alan öncelik bilgisi BT birimi için kritik öneme sahiptir. Ancak, talebi giren kişilerin inisiyatifine bırakılan öncelik kararı her zaman gerçekçi olmamaktadır. Örneğin, kritik olmayan ve düşük öncelikli bir değişiklik talebi yüksek öncelikli olarak girilebilmekte, bu da hatalı planlama ve müşteri memnuniyetsizliği ile sonuçlanabilmektedir. Bu çalışmada, iç müşteri talepleri metin madenciliği yöntemleriyle sınıflandırılarak taleplerin önem derecesi tahmin edilmeye çalışılmıştır. Sistemin eğitimi ve testi için kurumsal bir şirketin talep yönetim sisteminden alınan kayıtlar kullanılmıştır. Ham metin formundaki talep verisi üzerinde temizlik ve önişleme işlemlerinin ardından, doküman-terim matrisinin oluşturulmasında TF-IDF (Terim Frekansı – Ters Doküman Frekansı) ağırlıklandırma yönteminden yararlanılmıştır. Oluşturulan veri seti üzerinde çeşitli sınıflandırma algoritmaları test edilmiş ve en yüksek başarım %54.1 F-Skoru ile Sequential Minimal Optimization algoritmasıyla elde edilmiştir. Ayrıca, aşırı örnekleme yoluyla sınıfların dengeli hale getirildiği veri seti üzerinde ise en yüksek başarıma %74.5 F-Skoru değeri ile Random Forest algoritmasıyla ulaşılmıştır.

Anahtar kelimeler: Yazılım mühendisliği, Talep önceliklendirme, Yapay öğrenme, Metin sınıflandırma, Random forest

Abstract

In corporations, software issues and software change demands are forwarded to the Information Technology (IT) unit via a demand management system. The priority information in this system has critical importance to the IT unit. However, the priority decision that is left to the individuals who create the demand records may not always be realistic. For instance, a non-critical and low-priority demand may be created with the highest priority, and this may lead to faulty planning and eventually to customer dissatisfaction. In this work, internal customer demands were classified using text mining techniques and their priorities were predicted. The system was trained and tested with the records extracted from the demand management system of a corporation. After cleaning and preprocessing the raw textual demand data, TF-IDF (Term Frequency – Inverse Document Frequency) weighting scheme was used when creating the document-term matrix. Several classification algorithms were tested on the data set generated, and the highest performance was obtained by Sequential Minimal Optimization algorithm with 54.1% F-Score. In addition, on the dataset made balanced with oversampling technique, the highest performance was achieved by Random Forest algorithm with 74.5% F-Score.

Keywords: Software engineering, Demand prioritization, Machine learning, Text classification, Random forest.

1 Giriş

Rekabetin yoğun olduğu iş dünyasında, şirketler, müşterilerine sundukları hizmetlerin kalitesini en yüksek düzeyde tutmaya çalışmaktadır. Bu noktada, müşterilere doğrudan ya da dolaylı olarak sunulan yazılımların kalitesi büyük öneme sahiptir. Müşterinin dokunduğu noktalarda alınan hatalar ve fark edilen eksiklikler iç müşteri birimlerine bildirilir. İç müşteri birimleri de bu hata ve eksiklikleri ilgili Bilgi Teknolojileri (BT) birimine yazılım geliştirme talebi olarak iletirler. Küçük ölçekli şirketlerde bu süreç e-posta yoluyla ya da sözlü olarak yürütülebilmektedir. Ancak, büyük ölçekli şirketlerde bu taleplerin uygun şekilde kayıt altına alınması ve tüm süreç boyunca izlenmesi hem şirket iş süreçleri hem de hizmet kalitesi açısından bir zorunluluktur. Bunun için bu talepler genellikle Talep Yönetim Sistemi olarak adlandırılan sistemler üzerinden oluşturularak ilgili birime iletilirler. Bu sistemde yer alan öncelik bilgisi BT birimi için kritik öneme sahiptir. Mevcut talepler öncelik sırasına göre işleme alınarak çözümlenmeye çalışılır. Ancak, iç müşteri birimlerinin, açtıkları taleplere kendi işlerinin çabuk halledilmesi için genellikle "yüksek" öncelik verdiği gözlenmektedir. Önem derecesi gerçekçi seçilmediği için bu talepler, gerçek "yüksek" öncelikli taleplerin önüne geçebilmektedir. Bu da hatalı süreç ve kaynak planlamasının yanı sıra müşteri memnuniyetsizliği ile sonuçlanabilmektedir [1]. Dolayısıyla, yazılım geliştirme taleplerinin önceliklerinin doğru belirlenmesi büyük önem taşımaktadır.

Özellikle son yıllarda bu konuda çeşitli çalışmaların yapıldığı ve bu çalışmaların genellikle hata raporlarının (bug report) önceliklendirilmesi ya da bu hataların önem (severity) derecelerinin tahmin edilmesi şeklinde olduğu görülmektedir. Bu çalışmaların genelinde, çeşitli açık-kaynak yazılımların hata veritabanlarındaki hata kayıtlarının metin bölümleri, metin madenciliği yöntemleriyle işlenmekte ve öncelik ve/veya önem dereceleri çeşitli popüler sınıflandırma algoritmalarıyla tahmin edilmeye çalışılmaktadır [2]-[5]. Hata kayıtlarındaki hataya dair metinsel açıklamaların yanı sıra hatanın yığın izlemesi (stack trace), hatanın yaşandığı platform ve işletim sistemi, hatanın tespit edildiği yazılım bileşeni/modülü gibi çeşitli kategorik bilgilerin de sınıflandırma modelinin oluşturulmasına dâhil edilerek sınıflandırma başarımının

arttırılmaya çalışıldığı çeşitli çalışmalar da bulunmaktadır [6]-[10].

Bu çalışmada ise, yalnızca hatalar değil, diğer çeşitli geliştirme isteklerini de içeren iç müşteri talepleri, metin madenciliği yöntemleriyle sınıflandırılarak taleplerin öncelikleri tahmin edilmeye çalışılmıştır. Bu sayede, BT birimlerine iç müşteri taleplerinin önceliklerinin değerlendirilmesi ve doğru planlama yapılması konusunda destekleyici bir sistem geliştirilmesi amaçlanmıştır. Bilindiği kadarıyla, yazılım geliştirme taleplerinin metin madenciliği kullanılarak önceliklendirilmesine yönelik, Türkçe dilinin yapısal zorluklarını dikkate alan bir çalışma literatürde bulunmamaktadır.

Sistemin eğitimi ve testi için Türkiye'de faaliyet gösteren kurumsal bir şirketin talep yönetim sisteminden alınan gerçek talep kayıtları kullanılmıştır ve bu çalışmanın özgün yönlerinden biri de budur. Bu kayıtlarda, iç müşteri tarafından belirlenmiş hatalı öncelikler yerine BT birimi tarafından gerçekçi olarak revize edilmiş öncelik değerleri bulunmaktadır ve sistemin geliştirilmesi sırasında bu doğru öncelik değerleri dikkate alınmıştır. Ham metin formundaki talep verisi üzerinde temizlik ve önişleme işlemleri gerçekleştirilmiş, ardından, doküman-terim matrisinin oluşturulmasında TF-IDF (Terim Frekansı-Ters Doküman Frekansı) ağırlıklandırma yönteminden yararlanılmıştır [11]. Elde edilen veri seti üzerinde Naive Bayes [12], Naive Bayes Multinomial [13], Sequential Minimal Optimization [14], Random Forest [15] ve Rotation Forest [16] gibi çeşitli sınıflandırma algoritmaları test edilmiş ve en yüksek başarıma %74.5 F-Skoru değeri ile Random Forest algoritmasıyla ulaşılmıştır.

Makalenin devam eden bölümleri şu şekilde organize edilmiştir: İkinci bölümde kullanılan veri seti, metin önişleme teknikleri, sınıflandırma algoritmaları ile değerlendirme ölçütlerinden bahsedilmiştir. Üçüncü bölümde, yapılan deneylerin sonuçları sunulmuştur. Dördüncü ve son bölümde ise elde edilen sonuçlar değerlendirilmiş ve gelecek çalışmalara dair önerilere yer verilmiştir.

2 Malzeme ve yöntem

2.1 Veri seti

Bu çalışmada, veri seti olarak özel bir şirketin yazılım geliştirme biriminde, iç müşteri taleplerinin takibi ve çözümü için kullanılmakta olan bir yazılımın veritabanından alınan talep kayıtları kullanılmıştır. Bu kayıtlardan ise yalnızca talep başlığı, açıklama ve öncelik sınıfı alanlarından yararlanılmış, diğer alanlar tamamen göz ardı edilmiştir. Kayıtlardaki TC Kimlik Numarası, Vergi Numarası ve çeşitli evrak numaraları gibi kişisel bilgiler temizlenmiştir. Verilerin veritabanından çekilmesi ve ön temizlik işleminin gerçekleştirilmesi için Visual Studio 2013 geliştirme ortamında C# programlama dili ile yazılan bir program kullanılmıştır. Bu program ile ayrıca kayıtların başlık ve açıklama alanları birleştirilerek, her bir kayıt ayrı bir metin dosyası olacak şekilde, öncelik sınıflarına göre ayrı bir klasöre kaydedilmiş ve metin madenciliğinin ilk adımı olan metin önişleme için hazır duruma getirilmiştir.

Tablo 1'de veri setinden örnek kayıtlar görülmektedir. Örnek kayıtlarda görüldüğü gibi metin alanların yazımında herhangi bir standart bulunmamakta ve kullanıcılar tarafından çokça yazım hatası yapılmaktadır. Bu durum genellikle tüm metin işleme ve metin madenciliği uygulamalarında karşılaşılan ve aşılması gereken önemli bir zorluktur.

Veri setindeki kayıtlar düşük, orta ve yüksek olmak üzere üç öncelik kategorisine dengesiz olarak dağılmış durumdadır. Veri setinde 43 düşük, 150 orta ve 151 yüksek öncelikli olmak üzere toplam 344 kayıt bulunmaktadır. Sınıflandırma çalışmalarında, modelin eğitimi için kullanılacak veri setindeki kayıtların ilgili sınıflara dengeli olarak dağılması, oluşturulan modelin başarımı bakımından önemlidir. Dengesiz sınıflarla oluşturulan modelin ürettiği tahminler, çoğunluk durumundaki sınıflara doğru meyilli olmakta ve bu da azınlık durumundaki sınıflardan örneklerin doğru tahmin oranının düşmesine neden olmaktadır.

Tablo 1: Veri setinden örnek kayıtlar.

Başlık	Açıklama	Öncelik
TARİH FORMATINDA HATA	BT SİSTEM LİSTE SORGULAR/POLİÇE HASAR LİSTESİ YENİ alanında tarih aralıkları ile rapor alındığında exel'deki tarih formatlarında hata olmakta ve düzeltilememekte düzeltilmesi için desteğinizi rica ederiz.	Düşük
SAĞLIK REASÜRANS RAPORU REASÜRANS KOLONU YUVARLAMA HK.	SAĞLIK REASÜRANS RAPORU REASÜRANS KOLONUNDAKİ TUTARLAR LE OLUŞLAN FİŞLERDEKİ TUTARLAR ARASINDA KURUŞ FARKLARI BULUNMAKTADIR. KONTROLÜNÜ VE RAPORUN FİŞE YANSIYAN ŞEKLİ İLE DÜZENLENMESİNİ RİCA EDERİZ.	Orta
Acente Tecdit Listesi	Merhaba, Acente tecdit listesinden bölge kodu olarak Bankasürans departmanı görünmemektedir. İlgili hatanın giderilmesi için desteğinizi rica ederim. Saygılarımla,	Orta
346 – M******* Kasko baz update	K03 G04 T 59,535 Binde 59.535 olan bazın 54 olarak acilen düzeltilmesi gerekmektedir. Teşekkürler	Yüksek

Bundan dolayı, bu çalışmada, azınlık durumunda kalan düşük öncelikli sınıfına ait örnek sayısı, aşırı örnekleme (oversampling) yapılarak diğer sınıflardaki örnek sayısına yaklaştırılmış ve dengeli veri seti ile de deneyler yapılarak sonuca etkisi araştırılmaya çalışılmıştır [17]. Aşırı örneklemede temel yaklaşım, azınlık durumundaki sınıftaki kayıtların çoğaltılmasıdır. Bunun için literatürde çeşitli yöntemler önerilmiş olmakla birlikte, bu çalışmada, azınlık durumundaki düşük öncelikli sınıfına ait 43 kaydın her birinin ikişer kopyası oluşturularak, bu sınıfa ait kayıt sayısı 129'a çıkartılmıştır. Ayrıca, sınıflardaki kayıt sayısının tam olarak eşit olması adına, orta ve yüksek öncelikli sınıflardaki kayıtlardan bazılarının rastgele olarak veri setinden çıkartılması ve böylece bu sınıflardaki kayıt sayısının da 129'a düşürülmesi yoluna gidilmiştir.

2.2 Metin önişleme

Ham metin formunda yani yapısal olmayan veri üzerinde veri madenciliği tekniklerinin uygulanabilmesi için metin verisinin öncelikle yapısal bir forma dönüştürülmesi gereklidir. Bu dönüşüm işlemi metin önişleme olarak adlandırılır. Bu çalışmada, önişleme için PRETO aracından yararlanılmıştır [18]. PRETO, özellikle Türkçe metinler üzerinde yüksek performansla önişleme yapmak üzere geliştirilmiş, açık kaynaklı bir yazılımdır. Kök bulma, durdurma sözcükleri filtreleme, n-gram oluşturma ve terim ağırlıklandırma gibi doğal dil işleme yeteneklerinin yanı sıra Doküman-Terim (D-T) Matrisinin çeşitli dosya formatlarında kaydedilebilmesi gibi kullanışlı özellikler de sunmaktadır.

Bu çalışmada, kök bulma işlevi için PRETO'daki Zemberek ve Ek Çıkaran (Affix Stripping) kök bulma seçenekleri kullanılarak ayrı ayrı deneyler yapılmıştır. Zemberek, eklemeli Türk dilleri için geliştirilmiş, açık kaynak kodlu, platformdan bağımsız bir doğal dil işleme kütüphanesidir [19]. Zemberek, sözlük tabanlı bir kök bulma yöntemi kullanmakta olup, tanımlı bir kök ve ek sözlüğü kullanarak bu işlemi yapmaktadır. Ek Çıkaran kök bulucu ise Türkçe'nin kural tabanlı yapısını kullanmaktadır ve eklerin sondan başa doğru çıkarılması yaklaşımıyla kelimenin kökünü bulacak şekilde geliştirilmiş bir doğal dil işleme yöntemidir [20]. Çalışmada kullanılan diğer bir önişleme tekniği ise durdurma sözcükleri filtrelemedir. PRETO ile birlikte gelen, Türkçe için belirlenmiş 181 durak sözcüğünü içeren bir liste kullanılmıştır [18]. Terim ağırlıklandırma yöntemi olarak, metin işleme alanında bilinen en etkin yöntemlerden biri olan TF-IDF kullanılmıştır. Ayrıca, önişleme sonucunda uzunluğu üç karakterden az olan sözcükler ile rakamlar ve noktalama işaretleri tamamen filtrelenmiştir. Terim oluşturma yöntemi olarak 1-gram ve 2-gram'ların sonuçlar üzerindeki etkilerini ayrı ayrı gözlemlemek için deneyler yapılmıştır.

2.3 Sınıflandırma

Bu çalışmada, kullanıcıların metin formunda girdiği taleplerin önceliklerinin tahmin edilmesi bir metin sınıflandırma problemi olarak ele alınmıştır.

Sınıflandırma, sınıfı bilinmeyen bir veri örneğinin, daha önce sınıfları bilinen veri örnekleriyle eğitilmiş bir model üzerinden sınıfının tahmin edilmesi olarak tanımlanır [12].

Literatürde çok sayıda sınıflandırma algoritması bulunmaktadır. Bu çalışmada ise çeşitli sınıflandırma algoritmaları denenmiş ve genel olarak etkinliği yüksek olanlar seçilerek deneyler çeşitlendirilmiş ve sonuçları sunulmuştur. Seçilen bu algoritmalar Naive Bayes, Naive Bayes Multinomial, Sequential Minimal Optimization, Random Forest ve Rotation Forest algoritmalarıdır.

2.3.1 Naive bayes (NB)

Naive Bayes algoritması, veri kümesindeki değerlerin kombinasyonunu ve frekansını dikkate alarak bir olasılık kümesi oluşturan temel bir istatistiksel olasılık sınıflandırıcıdır. Yapısının basitliği, hızlı çalışması ve karmaşık sınıflandırma algoritmalarıyla kıyaslanabilir sonuçlar üretmesi sayesinde sıklıkla tercih edilen bir algoritmadır [12].

2.3.2 Naive bayes multinomial (NBM)

Naive Bayes Multinomial algoritması, Naive Bayes algoritmasının metin belgeleri için özelleştirilmiş bir halidir. NB'den farklı olarak sözcükler için çok terimli dağılım kullanan NBM, metin sınıflandırmada NB'ye göre daha yüksek başarım sergilemektedir [13].

2.3.3 Sequential minimal optimization (SMO)

Destek Vektör Makineleri'nin (SVM - Support Vector Machines) eğitilmesi için çok büyük bir kuadratik programlama eniyileme probleminin çözülmesi gerekmektedir. SMO, bunu etkili bir şekilde gerçekleştiren, hızlı bir SVM eğitim algoritmasıdır [14].

2.3.4 Random forest (RaF)

Karar ağaçları, sınıflandırma görevlerinde yaygın olarak kullanılan bir öğrenme yöntemidir. Adından da anlaşılacağı gibi eğitilen model ağaç görüntüsüne benzer bir yapıda olup dal ve yapraklardan oluşur. Ağaçta bulunan her bir düğüm bir test koşulunu temsil edip, test sonucu oluşan daldaki yaprak, test edilen verinin sınıfını belirtir. Random Forest algoritmasında ise eğitim kümesindeki verilerden rastgele seçimler yapılarak çok sayıda karar ağacı oluşturulur [15]. Test verisi tüm ağaçlarda test edildikten sonra çoğunluğun kararı, ilgili verinin sınıfı olarak belirlenir.

2.3.5 Rotation forest (RoF)

Rotation Forest algoritması, eğitim kümesinden farklı öznitelik kombinasyonları kullanarak çok sayıda karar ağacı oluşturur [16]. Test verisi yine tüm ağaçlarda test edildikten sonra çoğunluğun kararı, ilgili verinin sınıfı olarak belirlenir.

2.4 Deney ortamı

Deney ortamı olarak, çeşitli sınıflandırma algoritmalarını bünyesinde barındıran ve deney sonuçlarının kolayca elde edildiği WEKA kullanılmıştır. WEKA, veri madenciliği çalışmaları için geliştirilmiş, Java tabanlı, açık kaynak kodlu bir araçtır [21]. PRETO aracından elde edilen doküman-terim matrisi dosyaları WEKA'da doğrudan kullanılamamaktadır. Bu nedenle, C# programlama dili kullanılarak geliştirilen bir program aracılığıyla, doküman-terim matrisi dosyaları, WEKA'nın ARFF dosya formatına dönüştürülmüştür.

Deneylerde kullanılan sınıflandırıcılar, WEKA'daki varsayılan parametreleriyle çalıştırılmıştır. Veri setindeki örnek sayısının azlığı nedeniyle tüm deneylerde 10-kat çapraz geçerleme uygulanmıştır.

2.5 Değerlendirme ölçütleri

Yapılan sınıflandırma deneylerinde, sınıflandırıcı başarımı için Kesinlik (Precision), Anımsama (Recall), F-Skoru (F-Measure)

ve ROC Alanı (Receiver Operating Characteristic Area) ölçütleri kullanılmıştır. Bu ölçütlerin tümünde ölçüt değeri ne kadar yüksek ise sınıflandırma başarımı o kadar yüksek demektir. Sınıflandırma modelinin test edilmesi sonucunda Doğru Pozitif (DP-True Positive), Yanlış Pozitif (YP-False Positive), Doğru Negatif (DN-True Negative) ve Yanlış Negatif (YN-False Negative) olmak üzere dört farklı çıktı elde edilir. Bu çıktılara bağlı olarak çeşitli başarım ölçütleri hesaplamak olasıdır. Kesinlik ve Anımsama değerleri Denklem (1) ve (2)'deki gibi hesaplanır.

$$Kesinlik = \frac{DP}{DP + YP} \quad (1)$$

$$Anımsama = \frac{DP}{DP + YN} \quad (2)$$

F-Skoru, kesinlik ve anımsamanın harmonik ortalamasıdır ve Denklem (3)'teki gibi hesaplanır.

$$F\text{-}Skoru = \frac{2 \times Kesinlik \times Anımsama}{Kesinlik + Anımsama} \quad (3)$$

ROC Alanı ölçütü, sınıflandırıcı başarımı değerlendirmede kullanılan etkili bir ölçüttür ve Denklem (4)'teki gibi hesaplanır.

$$ROC\ Alanı = \frac{1}{2}\left(\frac{DP}{DP + YN} + \frac{DN}{DN + YP}\right) \quad (4)$$

3 Bulgular

Farklı önişleme seçenekleri uygulanarak elde edilen sekiz farklı doküman-terim matrisi üzerinde Bölüm 2.3'te bahsedilen sınıflandırma algoritmaları çalıştırılarak sınıflandırma başarımları gözlenmiştir. Tablo 2'de bu doküman-terim matrislerinin özellikleri özetlenmiştir.

Tablo 2: Doküman terim matrislerinin özellikleri.

	Sınıftaki Doküman Sayısı			Önişleme Seçenekleri	
No	Düşük	Orta	Yüksek	Kök Bulucu	n-gram
1	43	150	151	Ek Çıkaran	1-gram
2	43	150	151	Ek Çıkaran	2-gram
3	43	150	151	Zemberek	1-gram
4	43	150	151	Zemberek	2-gram
5	129	129	129	Ek Çıkaran	1-gram
6	129	129	129	Ek Çıkaran	2-gram
7	129	129	129	Zemberek	1-gram
8	129	129	129	Zemberek	2-gram

Sekiz doküman-terim matrisi üzerinde beş sınıflandırma algoritması işletilerek toplam 40 deney sonucu elde edilmiş olup, bu sonuçlar Tablo 3 ve 4'te toplu halde gösterilmektedir. Tablolarda her bir doküman-terim matrisi üzerinde elde edilen en yüksek başarım değerleri koyu olarak gösterilmiştir.

Dengesiz sınıflara sahip veri seti ile elde edilen sonuçlara göre en yüksek F-Skoru ve ROC Alanı değerleri 1 No.lu doküman-terim matrisinde sırasıyla 0.541 ve 0.690 olarak bulunmuştur. Ek Çıkaran kök bulucu ve 1-gram önişleme seçeneklerinin genellikle daha iyi sonuç verdiği görülmektedir. Diğer önişleme seçeneklerinde de karar ağacı tabanlı sınıflandırıcılar olan Random Forest ve Rotation Forest algoritmalarının yüksek başarım gösterdikleri görülmektedir.

Dengeli sınıflara sahip veri seti ile elde edilen sonuçlara göre en yüksek F-Skoru ve ROC Alanı değerleri 5 No.lu doküman-terim matrisinde sırasıyla 0.745 ve 0.898 olarak bulunmuştur. ROC Alanı ölçütüne göre tüm önişleme seçeneklerinde en yüksek başarım yine karar ağaçları tabanlı algoritmalarla elde edilmiş olup Random Forest algoritması en yüksek başarıma sahiptir. Diğer ölçütler bakımından ise SMO algoritmasıyla elde edilen yüksek sonuçlar da dikkat çekicidir.

Tablo 3: Dengesiz sınıflardan oluşan doküman-terim matrislerinde deney sonuçları.

D-T Matrisi	Sınıflandırıcı	Kesinlik	Anımsama	F-Skoru	ROC Alanı
1	NB	0.485	0.488	0.486	0.570
	NBM	0.533	0.535	0.532	0.665
	SMO	**0.542**	0.541	**0.541**	0.630
	RaF	0.496	**0.561**	0.526	**0.690**
	RoF	0.490	0.523	0.501	0.653
2	NB	**0.573**	**0.520**	**0.527**	**0.660**
	NBM	0.486	0.462	0.467	0.643
	SMO	0.506	0.517	0.507	0.597
	RaF	0.528	0.515	0.519	0.677
	RoF	0.497	0.520	0.496	0.650
3	NB	0.456	0.456	0.455	0.577
	NBM	0.465	0.465	0.465	0.580
	SMO	0.489	0.491	0.490	0.580
	RaF	**0.556**	**0.541**	**0.512**	**0.670**
	RoF	0.494	0.523	0.498	0.628
4	NB	**0.494**	0.468	0.474	0.627
	NBM	0.449	0.422	0.429	0.606
	SMO	0.467	0.480	0.469	0.550
	RaF	0.481	0.488	0.484	0.625
	RoF	0.485	**0.532**	**0.496**	**0.645**

Tablo 4: Dengelenmiş sınıflardan oluşan doküman-terim matrislerinde deney sonuçları.

D-T Matrisi	Sınıflandırıcı	Kesinlik	Anımsama	F-Skoru	ROC Alanı
5	NB	0.611	0.612	0.610	0.767
	NBM	0.718	0.718	0.717	0.849
	SMO	0.712	0.724	0.716	0.817
	RaF	**0.747**	**0.749**	**0.745**	**0.898**
	RoF	0.687	0.700	0.690	0.878
6	NB	0.601	0.592	0.560	0.825
	NBM	0.681	0.695	0.686	0.872
	SMO	**0.730**	**0.729**	**0.727**	0.833
	RaF	0.724	0.724	0.723	**0.880**
	RoF	0.696	0.700	0.698	0.866
7	NB	0.569	0.563	0.565	0.745
	NBM	0.624	0.612	0.617	0.791
	SMO	0.631	0.651	0.637	0.766
	RaF	**0.721**	**0.726**	**0.721**	**0.886**
	RoF	0.656	0.672	0.660	0.858
8	NB	0.573	0.592	0.570	0.800
	NBM	0.657	0.669	0.662	0.851
	SMO	**0.692**	**0.693**	**0.692**	0.810
	RaF	0.678	0.680	0.679	**0.853**
	RoF	0.667	0.667	0.664	0.829

5 No.lu doküman-terim matrisinde Random Forest algoritmasıyla 10-kat çapraz geçerleme sonucunda elde edilen başarım değerlerinin üretildiği karmaşıklık matrisi (confusion matrix) Tablo 5'te görülmektedir. Karmaşıklık matrisine göre, düşük öncelikli 129 talebin tamamı yine düşük öncelikli olarak eksiksiz bir biçimde tahmin edilebilmiştir. Ancak, gerçekte orta öncelikli olan taleplerin 91 tanesi orta, 34 tanesi yüksek ve 4 tanesi ise düşük öncelikli olarak tahmin edilmiştir. Benzer şekilde, gerçekte yüksek öncelikli olan taleplerin 70 tanesi

yüksek, 56 tanesi orta ve 3 tanesi düşük öncelikli olarak tahmin edilmiştir.

Tablo 5: 5 No.lu D-T matrisinde Random Forest algoritmasının çalıştırılmasıyla elde edilen karmaşıklık matrisi.

		Tahmin Edilen Sınıf			
		Düşük	Orta	Yüksek	Toplam
Gerçek Sınıf	Düşük	**129**	0	0	129
	Orta	4	**91**	34	129
	Yüksek	3	56	**70**	129
	Toplam	136	147	104	387

Görüldüğü gibi düşük öncelikli talepler açıkça diğerlerinden ayrılabilirken, orta ve yüksek öncelikli taleplerin ayrıştırılmasında başarı oranı düşmektedir. Bu durumun oluşmasındaki temel sebebin, farklı öncelik sınıflarına ait talep kayıtlarında geçen sözcüklerin sıklığı ve dağılımı olduğu düşünülmektedir. Tablo 6'da her bir öncelik sınıfındaki taleplerde en sık geçen ilk 20 terim (ön işlemeden sonra doküman-terim matrisinde yer aldığı ve sınıflandırıcıların eğitiminde kullanıldığı haliyle) ve bunların sıklıkları verilmiştir. Tabloda koyu yazı tipi ile yazılan terimler, yalnızca ilgili öncelik sınıfında bulunan ve diğer sınıflarda yer almayan (ilk 20 içerisinde) terimlerdir. Tablo 6'daki değerler incelendiğinde, orta ve yüksek öncelikli taleplerde sıklıkla aynı sözcüklerin kullanıldığı, düşük öncelikli taleplerde ise sözcük kullanımının ve sözcüklerin sıklık sıralarının farklılaştığı görülmektedir.

Tablo 6: Sınıflardaki en sık ilk 20 terim.

	Düşük		Orta		Yüksek	
Sıra	Terim	Sıklık	Terim	Sıklık	Terim	Sıklık
1	zeyil	60	rica	73	poliç	101
2	dosya	42	eder	67	eder	79
3	poliçe	39	merhap	55	rica	76
4	**rapor**	39	poliçe	43	merhap	67
5	rica	36	poliç	43	nol	56
6	siste	36	nol	34	acil	47
7	poliç	36	iptal	34	poliçe	46
8	ace	36	dosya	32	kayıt	46
9	**tarih**	36	**transfer**	29	iptal	40
10	eder	33	siste	27	siste	36
11	**uyar**	27	acil	25	kontrol	31
12	kontrol	24	kayıt	25	zeyil	31
13	hatas	24	zeyil	25	hatas	30
14	**işle**	24	kontrol	24	**teklif**	30
15	nol	21	imagı	24	**prim**	28
16	**not**	21	png	24	**aşağı**	26
17	**provizyo**	21	thumbnail	24	**hata**	25
18	**limit**	21	ace	23	imagı	25
19	merhap	18	hatas	22	png	25
20	acil	18	**sistem**	18	thumbnail	25

Orta ve yüksek öncelik sınıflarındaki 20 terimden 16'sı ortak olmakla birlikte bunların sıklık sıraları da çoğunlukla birbirine yakındır. Bu durum, bu iki sınıftan talep kayıtlarının ayrıştırılmasında başarı oranının düşük olmasına neden olmaktadır. Buna karşın, düşük öncelik sınıfındaki 20 terimden 13'ü orta ve 11'i yüksek öncelik sınıfındaki terimlerle ortak olmakla birlikte terim sıklık sıraları da önemli ölçüde farklılık göstermektedir. Örneğin, düşük öncelik sınıfındaki en sık *zeyil* teriminin orta ve yüksek öncelik sınıflarındaki sıklık sıraları 13 ve 12'dir. Benzer şekilde, düşük öncelik sınıfında en alt sıradaki *merhap* ve *acil* terimleri ise diğer sınıfların sıklık sıralamasında üst sıralarda yer almaktadır. Bu belirgin farklılık, düşük öncelikli taleplerin orta ve yüksek öncelikli taleplerden ayrıştırılmasındaki yüksek başarı oranını açıklamaktadır.

Sınıflandırıcı başarımı bakımından elde edilen sonuçlar genel olarak çok şaşırtıcı değildir çünkü Random Forest gibi topluluk öğrenmesi yöntemlerinin bireysel yöntemlere göre daha yüksek sınıflandırma başarımı sergiledikleri bilinmektedir. Yine bir topluluk öğrenmesi yöntemi olan Rotation Forest algoritmasının da Random Forest algoritmasına yakın sonuçlar ürettiği deney sonuçlarında görülmektedir.

4 Tartışma ve sonuç

Bu çalışmada, kullanıcıların metin formunda girdiği yazılım geliştirme taleplerinin önceliklerinin tahmin edilmesi bir metin sınıflandırma problemi olarak ele alınmıştır. Yüksek başarımları nedeniyle metin sınıflandırma uygulamalarında sıkça kullanılan çeşitli algoritmalar, farklı önişleme seçenekleriyle elde edilmiş doküman-terim matrisleri üzerinde çalıştırılarak elde edilen sonuçlar karşılaştırılmıştır. Mevcut veri setindeki kayıtların öncelik sınıfları üzerindeki dağılımının dengesiz olması nedeniyle sınıf dengeleme yoluna gidilmiş, dengesiz ve dengeli veri setleriyle deneyler yapılarak elde edilen sonuçlar sunulmuştur.

Deney sonuçlarına göre Random Forest algoritmasının genellikle tüm koşullarda iyi sonuçlar verdiği görülmektedir. Dengeli veri seti üzerinde Random Forest algoritmasıyla 0.745 F-Skoru ve 0.898 ROC Alanı sonuçlarına ulaşılmıştır. Yine karar ağacı tabanlı bir algoritma olan Rotation Forest ile de yakın sonuçlar elde edilmiştir.

Bu çalışmada ele alınan problemin yaşandığı şirketin talep yönetim sistemine yapılacak bir eklemeyle, talep önceliklendirme konusunda BT birimine yardımcı olacak bir karar destek sisteminin oluşturulması ve işlerliğinin denenmesi planlanmaktadır. Bu iyileştirme çalışması kapsamında, iç müşterinin belirlediği ilk öncelik değerinin ve bu sistemin tahmin ettiği öncelik değerinin de kayıtlar üzerinde saklanması ve böylece sistem başarımının izlenebilir olması hedeflenmektedir. Ayrıca, saklanacak bu bilgilerin, sistemin yeniden eğitimi ve hassas ayarlamalar yapılabilmesi bakımından yararlı olacağı değerlendirilmektedir.

Bu çalışmada, kullanılan veri setinin küçük olmasının sınıflandırma başarımını düşüren bir etken olduğu düşünülmektedir. Veri setinin elde edildiği talep yönetim sisteminden ileride elde edilecek daha geniş bir veri seti ile benzer deneyler yapılarak başarımın arttırılması sağlanabilir. Ek olarak, çeşitli önişleme ve ağırlıklandırma teknikleri ile Yapay Sinir Ağları ve son zamanlarda oldukça etkili sonuçlar verdiği görülen Derin Öğrenme yöntemleri kullanılarak daha iyi bir talep önceliklendirme yapılmaya çalışılabilir.

5 Kaynaklar

[1] Uddin J, Ghazali R, Deris MM, Naseem R, Shah H. "A survey on bug prioritization". *Artificial Intelligence Review*, 47(2), 145-180, 2017.

[2] Tian Y, Lo D, Sun C. "Information Retrieval Based Nearest Neighbor Classification for Fine-Grained Bug Severity Prediction". *19th Working Conference on Reverse Engineering,* Ontario, Canada, 15-18 October 2012.

[3] Sharma M, Bedi P, Chaturvedi KK, Singh VB. "Predicting the priority of a reported bug using machine learning techniques and cross project validation". *12th International Conference on Intelligent Systems Design and Applications (ISDA)*, Kochi, India, 27-29 November 2012.

[4] Sharma G, Sharma S, Gujral S. "A novel way of assessing software bug severity using dictionary of critical terms". *Procedia Computer Science*, 70, 632-639, 2015.

[5] Zhang T, Chen J, Yang G, Lee B, Luo X. "Towards more accurate severity prediction and fixer recommendation of software bugs". *Journal of Systems and Software*, 117, 166-184, 2016.

[6] Kanwal J, Maqbool O. "Bug prioritization to facilitate bug report triage". *Journal of Computer Science and Technology*, 27(2), 397-412, 2012.

[7] Kaushik N, Amoui M, Tahvildari L, Liu W, Li S. "Defect Prioritization in the Software Industry: Challenges and Opportunities". *IEEE 6th International Conference on Software Testing, Verification and Validation*, Luxembourg, Luxembourg, 18-22 March 2013.

[8] Alenezi M, Banitaan S. "Bug Reports Prioritization: Which Features and Classifier to Use?". *12th International Conference on Machine Learning and Applications*, Florida, USA, 4-7 December 2013.

[9] Yang C, Chen K, Kao W. "Improving severity prediction on software bug reports using quality indicators". *IEEE 5th International Conference on Software Engineering and Service Science*, Beijing, China, 27-29 June 2014.

[10] Tian Y, Lo D, Xia X, Sun C. "Automated prediction of bug report priority using multi-factor analysis". *Empirical Software Engineering*, 20(5), 1354-1383, 2015.

[11] Schütze H, Manning CD, Raghavan P. *Introduction to Information Retrieval*. New York, USA, Cambridge University Press, 2008.

[12] Han J, Kamber M. *Data Mining: Concepts and Techniques*. 2nd ed. California, USA, Morgan Kaufmann Publishers, 2006.

[13] Kibriya AM, Frank E, Pfahringer B, Holmes G. "Multinomial naive bayes for text categorization revisited". *17th Australian joint conference on Advances in Artificial Intelligence*, Cairns, Australia, 4-6 December 2004.

[14] Platt JC. *Fast Training of Support Vector Machines Using Sequential Minimal Optimization*. Editors: Bernhard S, Christopher JCB, Alexander JS. Advances in Kernel Methods, 185-208, Massachusetts, USA, MIT Press, 1999.

[15] Breiman L. "Random forests". *Machine Learning*, 45(1), 5-32, 2001.

[16] Rodriguez JJ, Kuncheva LI, Alonso CJ. "Rotation forest: A new classifier ensemble method". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10), 1619-1630, 2006.

[17] Rahman MM, Davis DN. "Addressing the class imbalance problem in medical datasets". *International Journal of Machine Learning and Computing*, 3(2), 224-228, 2013.

[18] Tunalı V, Bilgin TT. "PRETO: A High-performance Text Mining Tool for Preprocessing Turkish Texts". *International Conference on Computer Systems and Technologies*, Ruse, Bulgaria, 22-23 June 2012.

[19] Akın AA, Akın MD, "Zemberek, an open source NLP framework for Turkic Languages", 2007.

[20] Eryiğit G, Adalı E. "An affix stripping morphological analyzer for Turkish". *International Conference on Artificial Intelligence and Applications*, Innsbruck, Austria, 16-18 February 2004.

[21] Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. "The WEKA data mining software: an update". *SIGKDD Explorations*, 11(1), 10-18, 2009.