Olay Kayıtları Ürün ve Platform Kodu Tespit Süreci Otomasyonu

Ethem Utku Aktaş[1], Aziz Göktepe[1], Gamze Pehlivan[1], Ümit Ülkem Yıldırım[1], Cemal Yılmaz[2]

[1] Arge Merkezi, Softtech A.Ş., İstanbul, Türkiye
{utku.aktas, aziz.goktepe, gamze.pehlivan, umit.yildirim}@softtech.com.tr
[2] Mühendislik ve Doğa Bilimleri Fakültesi, Sabancı Üniversitesi, İstanbul, Türkiye
cyilmaz@sabanciuniv.edu

Özet. Yazılım ürünleri ile ilgili sorunların kayıt altına alınarak takibi ve çözümlenmesi için çeşitli platformlar mevcuttur. Bu kayıtların, ilgili platformlar aracılığıyla, büyük bir yazılım şirketindeki hangi ekibe iletilmesi gerektiği, kaydın çözümünde vakit kaybına neden olabilen önemli bir operasyonel sorundur. Softtech A.Ş. sahipliğindeki yazılım ürünlerine ait sorunların takibi de bir olay kaydı takip platformu aracılığıyla yapılmaktadır. Gelen kayıtların girişleri teknik olmayan yardım masası çalışanlarınca yapılmakta ve ilgili yazılım ekibine yönlendirilmektedir. Kaydı açan çalışanlar kayıtla ilişkili yazılım ürün kodu ve platform kodu bilgisi girişlerini yapmakta, bu girişler kaydın hangi yazılım ekibine atanacağını belirlemektedir. Ancak giriş yapan kişilerin ürün kodu ve platform kodu ile ilgili detaylı bilgi sahipliği bulunmamasının yanında, hatalı giriş yapılması kaydın farklı ekipler arasında dolaşmasına ve geç çözülmesine neden olabilmektedir. Bu çalışma ile ürün kodu ve platform kodu girişlerinin otomasyonu hedeflenmiştir. Yapılan kapsamlı testler sonucunda %66 f-skor ve %67 doğruluk değerlerine ulaşılmıştır.

Anahtar kelimeler: metin madenciliği, sınıflandırma algoritmaları, yazılım olay takip.

The Automation of the Process of Determining Product and Platform Codes for Issue Records

Abstract. There are various platforms for tracking and solving issue records related to software products. It is an important operational issue in a large software company, to which team the record should be assigned on these platforms, which can lead to a waste of time in the solution of the record. The tracking of software products of Softtech A.S. are also done on such an issue tracking platform. The issues are entered by non-technical help-desk staff and forwarded to related software team. The help desk staff enter software product code and platform code for the records and this information is used in assigning the records to the related software teams. But the staff entering the issues don't have detailed information about the product and platform codes, and wrong entrance causes assigning the records to wrong teams. Since the record may circulate between different teams, this is thought to have an effect on the late

solution of records. In this study, it's aimed to automate the process of assigning product and platform codes for issue records. As a result of comprehensive tests, 66% f-score and 67% accuracy were obtained.

Keywords: text mining, classification algorithms, software issue tracking.

1 Giriş

Yazılım uygulamalarına ilişkin sorunları takip etmek için olay kayıtları veya hata raporu takip sistemleri kullanılmaktadır. Proje veya yazılım şirketi küçükse excel tabloları izleme ve takip için yeterli olabilmektedir. Ancak büyük bir proje veya şirket söz konusu olduğunda olay kaydı takip sistemlerine ihtiyaç duyulmaktadır.

Olay kayıtlarının takibi için çeşitli sistemler bulunmaktadır. Büyük yazılım sistemlerinde bu takibin zorluğunu anlamak için Bugzilla'yı inceleyebiliriz. Bugzilla, açık kaynaklı bir web tarayıcı projesi olan Mozilla ile ilgili oluşan sorunları takip etmek için kurulmuş bir olay kayıt takip sistemidir [1]. Bugzilla'nın gösterge panelini incelediğimizde [2], çözülmüş binlerce sorun olduğu gibi, ilgilenilmeyi ve çözülmeyi bekleyen de binlerce sorunun hala sırasını beklemekte olduğunu görmekteyiz.

Yeni bir sorun bildirildiğinde ilgili arayüz aracılığıyla kaydın girişi yapılır ve çözülmesi için ilgili ekibe veya geliştiriciye atanır. Bu atama işlemi çoğunlukla arayüzden yapılır ve şirket büyük olduğunda ve sistemde doğru yazılım ekibini tanımlayan birçok ürün ve platform bulunduğunda zaman alıcı ve sıkıcı bir görev olabilmektedir. Diğer bir sorunsa doğru ekip seçilmediğinde kaydın farklı ekipler arasında dolaşabilmesi ve bu durumun kaydın çözüm süresini uzatabilmesidir.

Softtech A.Ş.'nin hizmet verdiği ana sektör bankacılık sektörü olup sisteminde tanımlı çok sayıda farklı ürün ve platformlar bulunmaktadır. Örnek vermek gerekirse "Kredi Kartı" bir ürünü, "ATM" veya "İnternet Şube" ise platformu ifade etmektedir. Ürün ve platform kombinasyonu, kaydı çözecek olan sorumlu ekibi tanımlamakta, ancak kayıtla ilgili doğru ürün kodu ve platform kodunun tespitinde yukarıda anlatılana benzer sorunlar yaşanmaktadır. İlgili ürün ve platformlar kaydı açan BT yardım masası personeli tarafından arayüzden belirlenmektedir.

Bu çalışmada amaç, açılan olay kayıtları ile ilişkili ürün ve platform kodlarının belirlenmesi sürecinin otomasyonudur. Amaç yardım masası personeli tarafından arayüzden elle girilen ilgili kodları, konu ve açıklama kısımlarındaki metin bilgilerini kullanarak tahmin etmek olduğundan, ilgili metnin, metin madenciliği teknikleri ile ön işlemden geçirilmesi gerekmektedir. Ön işlemden geçirildikten sonra ürün ve platform kodlarını tahmin etmek için makine öğrenme algoritmaları kullanılmış ve sonuçları karşılaştırılmıştır.

Bu çalışmanın üç konuda katkısı bulunmaktadır:

1. Aynı anda tahmin edilmesi gereken iki ayrı sınıf olduğundan, bu iki sınıfın kombinasyonlarını tahmin etmek için üç farklı yaklaşım önerilmiş ve sonuçları karşılaştırılmıştır.
2. Bu sınıflar yalnızca konu bilgisi kullanılarak tahmin edilmiş, sonrasında hem konu hem de açıklama bilgisi kullanılarak tahmin edilmiş ve sonuçlar karşılaştırılmıştır.

3. Bu iki sınıfın belirlenmesi bir servis aracılığıyla yapılabilir hale gelmiştir. Bu servisin sisteme entegre edilmesi ile birlikte, arayüzden yürütülen operasyonel süreç ortadan kalkacak ve bu durum verimliliği artıracaktır.

2 İlgili Çalışmalar

Olay kayıtlarının atanması ile ilgili olarak bugüne kadar çeşitli çalışmalar yapılmıştır. Anvik, Hiew ve Murphy [3] yazılım geliştiricilerin çözümledikleri kayıt tiplerini öğrenmek için makine öğrenmesi algoritmaları uygulamışlar ve iki farklı proje için %57 ve %64 kesinlik (precision) seviyelerine ulaşmışlardır. Baysal, Godfrey ve Cohen [4] kullanıcıların mevcut iş yükü, deneyimleri, uzmanlıkları gibi bilgileri kullanarak kaydı uygun kullanıcıya otomatik olarak atayan bir sistem önermişlerdir. Matter, Kuhn ve Nierstrasz [5] yazılım geliştiricilerin kaynak koduna yaptıkları katkılar ile ilişkili sözcükleri ve olay kayıtlarındaki sözcük bilgisini kullanarak olay kaydı ile ilgili uzmanlığı olabilecek geliştiricileri tahminlemişlerdir. Bhattacharya, Neamtiu ve Shelton [6] kayıt atama sürecini otomatikleştirmek için makine öğrenme algoritmaları ve grafik temelli bir model kullanmış ve % 86.09 tahmin doğruluğu elde etmişlerdir.

Zhou ve arkadaşları [7], yaptıkları çalışmanın ilk kısmında, yalnızca konu bilgilerini kullanarak hata raporlarını sınıflandırmışlar, açıklama alanını bu sınıflandırmada girdi olarak kullanmamışlardır. Bir satırlık bilgi içeren konu alanının, raporların sınıflandırılmasında açıklama alanında geçen bilgi ile aynı temel bilgileri sağladığını önermişlerdir. Yaptığımız çalışmada, yalnızca konu bilgisini kullanarak sınıflandırma ve hem konu hem de açıklama bilgilerini kullanarak sınıflandırma kıyaslanmış ve yapılan önermenin kullandığımız veri için de geçerli olup olmadığı sınanmıştır.

Bu çalışmada tahmin için metin verisi kullanıldığından metin madenciliği tekniklerine ihtiyaç duyulmuştur. [8] ve [9] 'da, olay kayıtlarının analizinde, metinler, dizge parçalama (tokenization), durak kelimeleri çıkarma (stop-word list) ve kök bulma (stemming) işlemlerinden geçirilmiş ve bir vektör uzayı modeli elde edilmiştir. Benzer şekilde, çalışmamızda tüm kayıtlar aynı süreçten geçirilmiştir. Yazılım çalışması Python programlama dili ile yapılmış ve Türkçe kelimeler için de hazırlanmış Python kütüphaneleri [10] [11] kullanılmıştır.

TF-IDF, bir kelimenin, her bir kayıt için önemini belirten bir metriktir [9]: Burada TF terim frekansı (term frequency), IDF ise ters doküman frekansıdır (inverse term frequency) ve aşağıdaki şekilde hesaplanır:

$$TF(t) = \text{Kayıtta t teriminin kaç kere geçtiği / Kayıttaki toplam terim sayısı} \quad [12]$$

$$IDF(t) = \log\_e(\text{Toplam kayıt sayısı / İçinde t terimi geçen kayıt sayısı}) \quad [12]$$

$$TF\text{-}IDF(t) = TF(t) * IDF(t) \quad [12]$$

Her bir kayıt için tf-idf vektörünü elde ettikten sonra makine öğrenme algoritmaları kullanılır ve performansları karşılaştırılır. Bu çalışmada K-NN, Lojistik regresyon, Sınıflama ve regresyon ağaçları (CART), Naive Bayes, Destek Vektör Makineleri (SVM) ve Çok Katmanlı Perseptron (MLP) algoritmaları seçilmiştir [16].

K-NN algoritmasında, tahmin edilmek istenen veriye en yakın mesafedeki, önceden seçilmiş K adet öğrenme verisinin sınıfına bakılarak bu yeni kaydın sınıfı tahmin

edilmektedir. Lojistik regresyon, ilgili verinin hangi sınıfa ait olduğunu olasılıksal olarak veren lineer bir modeldir. Sınıflama ve regresyon ağaçları (CART), ikili bir karar ağacı modeli ortaya koyarak sınıflandırma yapılabilmesini sağlayan bir algoritmadır. Naive Bayes, girdi olarak kullanılan değişkenlerin birbirinden bağımsız olduğunu varsayar ve bu değişkenlerin ilgili sınıfa ait olma olasılıkları üzerinden tahmin yapılmasını sağlar. Destek Vektör Makineleri (SVM) ile, bir düzlem üzerinde, ilgili sınıflar arasında bir sınır çizilmesi amaçlanır. Çok Katmanlı Perseptron (MLP), bir veya daha fazla gizli katman (hidden layer) içerebilen bir yapay sinir ağı modelidir.

Bu algoritmaların sonuçlarını karşılaştırmak için f-skor ve doğruluk (accuracy) metrikleri kullanılmıştır. F-skor, kesinlik (precision) ve duyarlılığın (recall) ağırlıklı harmonik ortalamasıdır. Kesinlik (precision), bir sınıf için doğru tahmin edilmiş kayıt sayısı, o sınıf için yapılan toplam tahmin sayısına bölünerek bulunmaktadır. Duyarlılık (recall) ise bir sınıf için doğru tahmin edilmiş kayıt sayısı, o sınıfa ait gerçek kayıt sayısına bölünerek bulunmaktadır [9]. Bu çalışmada tahmin edilen çok sayıda ürün ve platform kodu bulunmakta ve verilen değerler ortalama değerleri ifade etmektedir.

$$Kesinlik = TP\ (true\ positives) / (TP\ (true\ positives) + FP\ (false\ positives)) \quad [9]$$

$$Duyarlılık = TP\ (true\ positives) / (FN\ (false\ negatives) + TP\ (true\ positives)) \quad [9]$$

$$F\text{-}skor = 2 * kesinlik * duyarlılık / (kesinlik + duyarlılık) \quad [9]$$

$$Doğruluk\ (Accuracy) = (Doğru\ sınıflandırılan\ kayıt\ sayısı) / (Toplam\ kayıt\ sayısı) \quad [14]$$

K-NN algoritmasında, ilgili kayda en çok benzeyen K adet kaydın bulunmasında kosinüs benzerliği kullanılmıştır. Kosinüs benzerliğinin daha iyi sonuçlar verdiği, tekrar eden olay kayıtlarının tespiti konusunda yapılan bir çalışmada da [8] tespit edilmiştir. Hesaplaması aşağıdaki şekilde yapılmaktadır.

$$Benzerlik = cos(\theta) = (v1 \,.\, v2) / (|v1| * |v2|) \quad [15]$$

Tf-idf vektörlerinin elde edilmesinde, yukarıda belirtilen algoritmaların kullanımında ve sonuçların karşılaştırılmasında "Scikit-learn" Python kütüphaneleri kullanılmıştır [13].

3 Çözüm Yaklaşımları

3.1 Tahminlemeden Önce Ürün ve Platform Kodlarının Birleştirilmesi ve Tek Bir Sınıf için Tahmin Yapılması

İlk yaklaşımda, veri setimizdeki ürün ve platform kodları birleştirilir ve tahminleme elde edilen bu birleşim üzerinden yapılır. Yani bu yöntemde tek bir sınıf elde edilir ve tahminleme bu sınıf için yapılır. Bu yaklaşım ile öğrenme sonucu olarak her bir algoritma tek bir model üretir.

Algoritmaların performansı, tahmin edilen ürün kodu - platform kodu kombinasyonu ile gerçek ürün kodu - platform kodu kombinasyonunun karşılaştırılması ile yapılmıştır.

3.2 Ürün ve Platform Kodlarının Ayrı Ayrı Tahmin Edilmesi ve Tahminlerin Sonradan Birleştirilmesi

İkinci yaklaşımda, ürün kodu ve platform kodları ayrı ayrı tahmin edilerek her bir algoritma için iki ayrı model elde edilmiş ve bu iki tahminin sonucu birleştirilerek nihai tahmin elde edilmiştir.

Birinci yaklaşımda olduğu gibi algoritmaların performansları, ürün kodu ve platform kodu tahminlerinin kombinasyonu ve ilgili ürün ve platform kodlarının gerçek değerleri karşılaştırılarak kıyaslanır.

Her iki sınıf ayrı ayrı tahmin edildiğinden bu yaklaşımda ürün kodu ve platform kodu arasında bir ilişki bulunmadığı varsayılır ancak gerçekte durum böyle değildir. Ürün kodları ile ilişkili sadece sınırlı sayıda platform kodu olması mümkündür. Dolayısıyla bu yaklaşım sonucunda elde edilen bazı tahminler pratikte mümkün olmayan değerler olabilir.

3.3 Hiyerarşik Yaklaşım

İkinci yaklaşım sonucunda gerçekte mümkün olmayan değerlerin elde edilebiliyor olması üçüncü bir yaklaşımı ortaya çıkarmıştır. Bu yaklaşımda önce ürün kodları tahmin edilir. Daha sonra veri setindeki sadece tahmin edilen bu ürün koduna ait kayıtlar kullanılarak platform kodları tahmin edilir. Yani bu yaklaşım sonucunda ürün kodunu tahmin etmek için bir model elde edilir, platform kodunu tahmin etmek içinse elimizdeki veri setinde varolan ürün kodu sayısı kadar model elde edilir.

4 Deneysel Çalışmalar ve Sonuçların Analizi

4.1 Veri Analizi

Birisi öğrenmek için diğeri de modeli test etmek için olmak üzere iki ayrı veri setimiz bulunmaktadır. Her iki veri setinde de çözülmüş olay kayıtları bulunmakta ve ilgili kayıtlardaki ürün ve platform kodları kaydın çözümü anındaki değerleri içermektedir. (Eğer çözülmemiş bir olay kaydı hatalı bir ekibe atanmış durumdaysa ürün kodu ve platform kodları değiştirilerek ilgili ekibin değiştirilmesi sağlanmaktadır.) Aşağıdaki tablo öğrenme için kullanılan veri setinin özet bilgilerini içermektedir:

Tablo 1. Öğrenme için kullanılan veri setinin ilk durumu.

Veri Seti Özelliği	Adet
Kayıt Sayısı	31355
Farklı Ürün Kodu Sayısı	279
Farklı Platform Kodu Sayısı	39
Farklı Ürün-Platform Kombinasyonu Sayısı	543

Tablodan görüldüğü gibi tahmin edilmesi beklenen çok sayıda ürün ve platform kodunun bulunuyor olması veri setinin detay analizinin yapılmasını gerektirmiştir.

Her bir ürün-platform kodu kombinasyonunun veri setinde kaç kere geçtiği sayılmış, sonrasında aynı sayıda tekrar eden kombinasyon sayısı bulunmuştur. Örnek vermek gerekirse elimizdeki verinin içinde sadece bir kez geçen 127 adet farklı ürün-platform kodu kombinasyonu olduğu, maksimum sayıda tekrar eden kombinasyondan ise veri setinde 3584 adet olduğu tespit edilmiştir.

31355 kayıt içinde sadece bir kez geçen bir kombinasyonun tahmin edilmeye çalışılması anlamlı bulunmamış, tahmin edilmek istenen sınıf sayısının azaltılması ve verinin sadeleştirilmesi amacıyla az sayıda tekrar eden kombinasyonların veri setinden çıkarılmasına karar verilmiştir. Öğretme aşamasında veri setinin 10 parçaya bölünerek çapraz doğrulama yapılması (10-fold cross validation) ve her parçada birden daha fazla kayıt olması düşüncesiyle 20'den daha az sayıda tekrar eden ürün-platform kodu kombinasyonuları çıkarılmıştır. Sonuç olarak oluşan tablo aşağıda verilmiştir:

Tablo 2. 20'den daha az sayıda tekrar eden ürün-platform kodu kombinasyonları çıkarıldıktan sonra elde edilen veri setinin özeti.

Veri Seti Özelliği	Adet
Kayıt Sayısı	29645
Farklı Ürün Kodu Sayısı	123
Farklı Platform Kodu Sayısı	27
Farklı Ürün-Platform Kombinasyonu Sayısı	184

Öğrenme için kullanılacak olan veri seti analiz edildikten ve hangi kayıtların bu veri setinden çıkarılacağına karar verildikten sonra, test için kullanılmak üzere ayrılmış olan ve yakın tarihli kayıtları içeren ikinci veri seti analiz edilmiştir. Bu ikinci veri seti öğrenme için kullanılandan tamamen bağımsız, farklı bir zaman aralığı için temin edilmiş kayıtları içermektedir. Aşağıda ilgili veri için özet bilgiler bulunmaktadır:

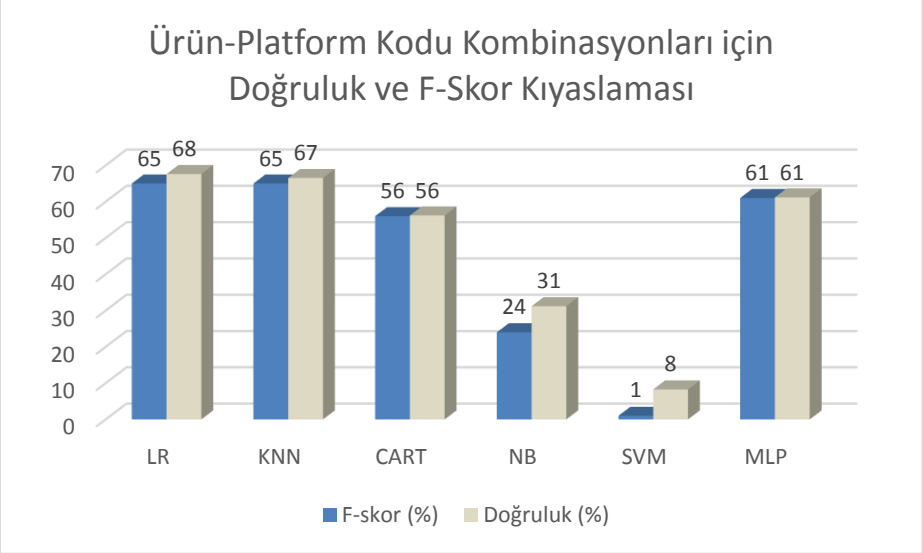
Tablo 3. Test veri seti özeti.

Veri Seti Özelliği	Adet
Kayıt Sayısı	824
Farklı Ürün Kodu Sayısı	104
Farklı Platform Kodu Sayısı	24
Farklı Ürün-Platform Kombinasyonu Sayısı	144

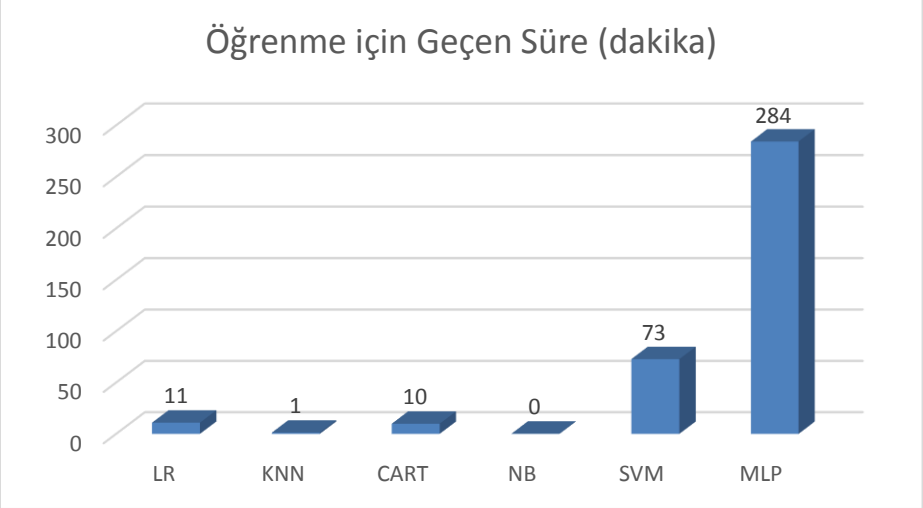
4.2 İlk Yaklaşım için Test Sonuçları

Öğrenme için kullanılacak olan veri setindeki ürün kodu ve platform kodları birleştirilmiş ve her bir algoritma için bir model elde edildikten sonra bu modeller kullanılarak test verisindeki ürün kodu – platform kodu kombinasyonları tahmin edilmiştir. Öğrenme aşamasında aşağıdaki algoritmalar kullanılmıştır: Lojistik Regresyon (LR), En Yakın K Komşu (KNN), Sınıflandırma ve Regresyon Ağaçları (CART), Naive Bayes (NB), Destek Vektör Makineleri (SVM) ve Çok Katmanlı Perseptron (MLP) (sadece bir adet gizli katman (hidden layer) ve ilgili katmanda 10 adet düğüm (node) olacak şekilde). Tahminlemede hem konu hem de açıklama

alanlarındaki bilgiler kullanılmıştır. Daha sonra her algoritma için oluşan model kullanılarak, test verisindeki ürün-platform kodu kombinasyonları tahmin edilmiştir. Aşağıdaki figürde f-skor ve doğruluk sonuçları, sonraki figürde ise öğrenme aşamasında ilgili algoritmaların benzer koşullarda ne kadar süre çalıştığı verilmiştir:



Şekil 1. Her bir algoritma için birinci yaklaşım sonucunda, ve hem konu hem de açıklama bilgileri kullanılarak elde edilen f-skor ve doğruluk kıyaslaması.



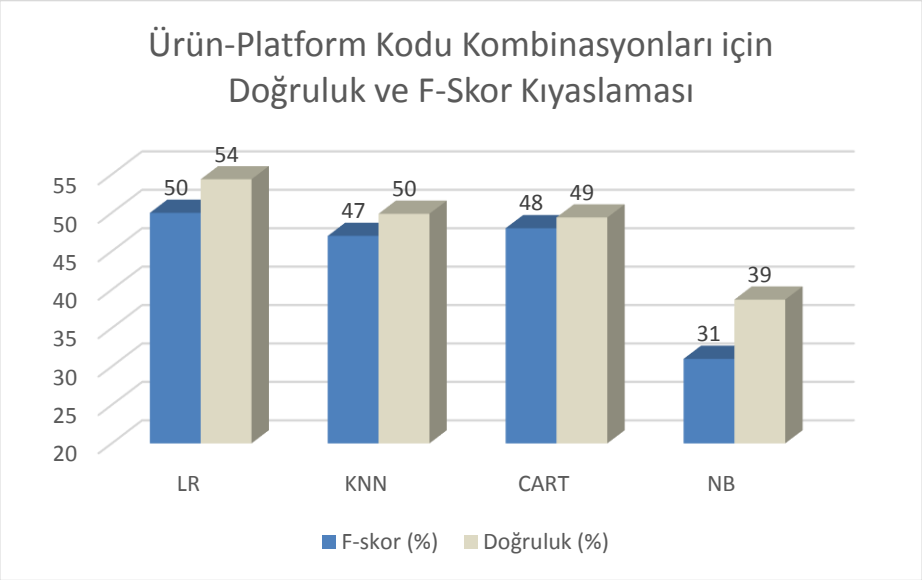
Şekil 2. Her bir algoritma için öğrenme için geçen sürenin kıyaslanması.

LR ve K-NN (K = 12) f-skor açısından en iyi sonuçları vermiştir (%65), LR aynı zamanda en iyi doğruluk sonucunu vermiştir (%68). SVM f-skor ve doğruluk açısından

yeterli olabilecek bir sonuç vermemiştir, MLP algoritmasının öğrenme aşaması çok uzun sürmüştür. Bu nedenlerden bu iki algoritma bundan sonraki testlerde değerlendirmeye dahil edilmemiştir.

K-NN algoritmasında en iyi sonucu verecek olan K değerini tespit etmek için algoritma, 1 ve 20 arasındaki tüm K değerleri denenerek ve sonuçlar kıyaslanarak denenmiş ve K'nın değeri 12 olduğunda en iyi sonucun alındığı görülmüştür.

Kayıtları sınıflandırmada bir satırlık konu bilgisinin, açıklama bilgisi ile aynı gerekli içeriğe sahip olduğu Zhou ve arkadaşlarının çalışmasında [7] önerilmiş ve bu çalışmada da sadece konu alanındaki bilgi kullanılarak tahminlemenin tekrarlanacağı belirtilmişti. Bu önermenin bizim veri setimiz için de geçerli olup olmadığı test edilmiştir. SVM ve MLP hariç tutularak testler tekrar edilmiş ve aşağıdaki sonuçlar elde edilmiştir:

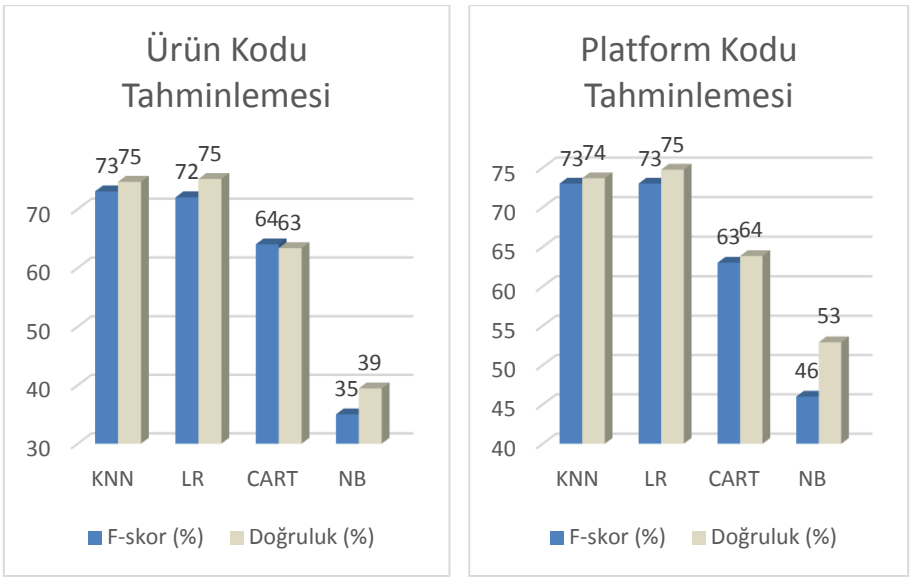


Şekil 3. Birinci yaklaşım ile, sadece konu bilgisini kullarak, algoritmaların f-skor ve doğruluk açısından kıyaslanması.

Yukarıdaki sonuçlardan görüleceği üzere kullanmakta olduğumuz veri için sadece konu bilgisinin kullanılması, konu ve açıklama bilgisinin kullanılması kadar iyi sonuç vermemiştir. LR, %50 f-skor ve %54 doğruluk değerleri ile en iyi sonucu vermiş, ancak sonuç olarak tahminlemede sadece konu bilgisinin kullanılması yaklaşımı daha sonraki testlerde tekrar edilmemiştir.

4.3 İkinci Yaklaşım için Test Sonuçları

Bu yaklaşımda, ürün kodları ve platform kodları ayrı ayrı tahmin edilmiş ve aşağıdaki sonuçlara ulaşılmıştır:



Şekil 4. Ürün ve platform kodlarının ayrı ayrı tahmin edilmesi sonuçları.

K-NN (K = 12) ürün kodunun tahminlemesinde f-skor (%73) açısından en iyi sonuçları verirken, doğruluk açısındansa LR ve K-NN (K = 12) en iyi sonuçları vermiştir (%75). Sadece platform kodunun tahminlemesinde ise LR doğruluk açısından en iyi sonucu verirken (%75), LR ve K-NN (K = 13, 15, 16) f-skor açısından en iyi sonucu vermiştir (%73). K-NN için en iyi sonucu verecek olan K değerinin bulunmasında 1 ile 20 arasındaki farklı K değerleri denenmiştir.

Sonuç olarak, ürün kodları ve platform kodları ayrı ayrı tahmin edildiğinde % 75 doğruluk oranına ulaşılmıştır. Ancak bu yaklaşımı diğer yaklaşımlarla kıyaslayabilmek için ayrı ayrı yapılan bu tahminler birleştirilmiş ve ürün–platform kodu kombinasyonlarının gerçek değerleri ile karşılaştırılmıştır. Eğer hem ürün hem de platform kodu doğru tahmin edilmişse tahminleme doğru olarak işaretlenmiş ancak herhangi birisinin hatalı olması durumunda, tahminleme hatalı kabul edilmiştir.

Ayrı ayrı yapılan tahminlerin birleştirilmesi ve bunların gerçek değerlerle karşılaştırılması sonrasında en iyi sonuçların ürün kodları K = 12 değeri ile K-NN algoritması ile tahminlendiğinde ve platform kodları K = 15 değeri ile yine K-NN algoritması ile tahminlendiğinde elde edildiği görülmüştür (f-skor = %66 ve doğruluk = %66).

4.4 Hiyerarşik Yaklaşım ile Elde Edilen Test Sonuçları

Bu yaklaşımda önce ürün kodları tahmin edilmiş (ikinci yaklaşım ile aynı şekilde) ve sadece bu ürün kodlarına sahip kayıtlar kullanılarak öğrenme algoritmaları çalıştırılmış ve platform kodları tahmin edilmiştir. Ürün kodlarının tahmininde ikinci yaklaşımda elde edilen sonuçlar elde edilmiştir (K = 12 değeri ile K-NN algoritması en iyi sonucu vermiştir). Birleştirilmiş ürün-platform kodu tahminleri, birleştirilmiş gerçek değerlerle kıyaslanmış (hem ürün kodu hem de platform kodu doğru olduğunda tahmin doğru sayılmıştır) ve ürün kodlarını tahminde K = 12 değeri ile K-NN algoritması, ve

sonrasında her bir ürün kodu için ayrı ayrı elde edilen modeller kullanılarak platform kodlarının tahmininde K = 15 değeri ile yine K-NN algoritması en iyi sonuçları vermiştir (f-skor = %66 ve doğruluk = %67).

4.5 Test Sonuçlarının Özeti

Aşağıdaki tabloda en iyi sonuçlar, bu sonuçları veren algoritma, ve f-skor ve doğruluk değerleri verilmiştir:

Tablo 4. Test Sonuçlarının Özeti.

Yaklaşım	Algoritma	F-Skor	Doğruluk
İlk yaklaşım: Birleştirilmiş tek sınıfın tahmin edilmesi	LR	%65	%68
İkinci yaklaşım: Ayrı ayrı tahmin	Ürün için K-NN (K = 12), Platform için K-NN (K = 15)	%66	%66
Hiyerarşik yaklaşım: Ürün kodunun ayrı tahmin edilmesi / Her ürün için sadece mümkün olan platform kodlarının tahmin edilmesi	Ürün için K-NN (K = 12), Platform için K-NN (K = 15)	%66	%67

Bu çalışmaya başlarken beklentimiz hiyerarşik yaklaşımın daha iyi sonuç vermesi yönündeydi, ancak yukarıdaki tablodan ilgili yaklaşımların benzer sonuçlar verdiği görülmektedir.

İkinci yaklaşım sonucunda elde edilen ürün kodu ve platform kodu kombinasyonlarının pratikte mümkün olmayan değerler olabilmesi nedeniyle ikinci yaklaşım tercih edilmemiştir. Ürün kodunu tek başına tahmin etmek %75 doğrulukla değerli bir bilgi verdiği için hiyerarşik yaklaşım tercih edilmiştir. Ancak yukarıdaki tablodan da görüleceği üzere hiyerarşik yaklaşım beklentimizin aksine bariz bir fark yaratmamıştır.

5 Sonuç ve Gelecek Çalışmalar

Bu çalışmada olay kayıtlarının ürün kodu ve platform kodu bilgisinin tahmin edilmesinde üç yaklaşım önerilmiş, test edilmiş ve sonuçlar karşılaştırılmıştır. Sonuç olarak hiyerarşik yaklaşım önerilmiştir. Bu yaklaşımda önce %75 doğruluk ve %73 f-skor değerleri ile ürün kodu tahmin edilmiş, daha sonra tahmin edilen bu ürün kodu bilgisi ile platform kodu tahmin edilmiştir. Sonuçta %66 f-skor ve %67 doğruluk değerlerine ulaşılmıştır. Bu değerler hesaplanırken hem ürün kodu hem de platform kodu doğru olduğunda tahminin doğru olduğu kabul edilmiştir.

Çözümlenmek istenen sınıflandırma probleminin diğer sınıflandırma problemlerinden ayırdedici özelliği aynı anda birden fazla sınıfın tahmin edilmek isteniyor olmasıdır. Çözüm önerisi olarak üç farklı yaklaşım sunulmuş, önerilen bu üç yaklaşım içinde hiyerarşik yaklaşımın daha iyi sonuçlar vermesi beklenirken tüm yaklaşımlar benzer sonuçlar vermiştir.

Hiyerarşik yaklaşım algoritmik olarak ve ortaya çıkan modellerin sayısı açısından daha karmaşık bir yaklaşımdır. Oysa birinci yaklaşım, yani birleştirilmiş olan tek sınıfın tahmin edilmesi algoritmik olarak daha basit bir yaklaşım olmasının yanında sadece tek bir model ortaya koymaktadır. Bu anlamda aynı anda birden fazla sınıfın tahmin edilmesi gereken böyle bir durumda sınıfların en başta birleştirilmesi yeterli görünmektedir. Ancak pratikte ürün kodunu tek başına tahmin etmek %75 doğrulukla değerli bir bilgi verdiği için hiyerarşik yaklaşım tercih edilmiştir.

Hazırlanan servisin sisteme entegrasyonu ile birlikte yardım masası personeli artık olay kaydı açarken ürün kodu ve platform kodu girişi yapmak zorunda olmayacaktır. Sadece konu ve açıklama bilgisi girilecek ve sistem ürün ve platform bilgisini otomatik olarak atayacaktır.

Sistemin otomasyonu ve oluşacak olan yeni veri ile öğrenme algoritmaları yeniden çalıştırıldığında; sistemin aynı ekibe kayıt atanmasına neden olacak alternatif ürün - platform kombinasyonları olması durumunda aynı kombinasyonu seçme eğilimi göstereceğini beklediğimizden, tahminlemedeki doğruluğun da artacağını tahmin etmekteyiz.

Yeni olay kayıtları açılırken bu kayıtlara en çok benzeyen geçmişte açılmış ve tamamlanmış olay kayıtları, bunların daha önce ne şekilde çözüldüğü bilgisi ile birlikte önerilecek şekilde çalışmalar devam edecektir. Ayrıca kayıtlar ekiplere atandıktan sonra kayıtların yazılım geliştiricilere atanması sürecinin de, geliştiricilerin iş yükü, uzmanlıkları gibi bilgiler de dikkate alınarak, otomasyonu için çalışmaların devam etmesi planlanmaktadır.

Kaynaklar

1. Serrano, N., Ciordia, I.: Bugzilla, ITracker, and other bug trackers. IEEE software 22.2 (2005): 11-13.
2. Bugzilla Homepage, https://bugzilla.mozilla.org, last accessed 2017/06/02.
3. Anvik, J., Hiew, L., Murphy, G. C.: Who should fix this bug?. Proceedings of the 28th international conference on Software engineering. ACM, 2006.
4. Baysal, O., Godfrey, M. W., Cohen, R.: A bug you like: A framework for automated assignment of bugs. Program Comprehension, 2009. ICPC'09. IEEE 17th International Conference on. IEEE, 2009.
5. Matter, D., Kuhn, A., Nierstrasz, O.: Assigning bug reports using a vocabulary-based expertise model of developers. Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on. IEEE, 2009.
6. Bhattacharya, P., Neamtiu, I., Shelton, C. R.: Automated, highly-accurate, bug assignment using machine learning and tossing graphs. Journal of Systems and Software 85.10 (2012): 2275-2292.
7. Zhou, Y., Tong, Y., Gu, R., Gall, H.: Combining text mining and data mining for bug report classification. Journal of Software: Evolution and Process (2016).
8. Runeson, P., Alexandersson, M., Nyholm, O.: Detection of duplicate defect reports using natural language processing. Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society, 2007.
9. Menzies, T., Marcus, A.: Automated severity assessment of software defect reports. Software Maintenance, 2008. ICSM 2008. IEEE International Conference on. IEEE, 2008.

10. PyPi Snowball-stemmer, https://pypi.python.org/pypi/snowballstemmer, last accessed 2017/06/03.
11. PyPi Stop-words, https://pypi.python.org/pypi/stop-words, last accessed 2017/06/03.
12. Tf-idf: A single page tutorial, http://www.tfidf.com, last accessed 2017/06/03.
13. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12.Oct (2011): 2825-2830.
14. Shihab, E., Ihara, A., Kamei, Y., Ibrahim, W. M., Ohira, M., Adams, B., Hassan, A. E., Matsumoto, K.: Predicting re-opened bugs: A case study on the eclipse project. Reverse Engineering (WCRE), 2010 17th Working Conference on. IEEE, 2010.
15. Jalbert, N., Weimer, W.: Automated duplicate detection for bug tracking systems. Dependable Systems and Networks with FTCS and DCC, 2008. DSN 2008. IEEE International Conference on. IEEE, 2008.
16. Scikit-learn: Machine learning in Python, http://scikit-learn.org/stable/supervised_learning.html, last accessed 2017/08/25