ÖHÜ Müh. Bilim. Derg. / OHU J. Eng. Sci.

ISSN: 2564-6605
doi: 10.28948/ngumuh.383709
Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi, Cilt 7, Sayı 1, (2018), 1-13
Omer Halisdemir University Journal of Engineering Sciences, Volume 7, Issue 1, (2018), 1-13

Araştırma / Research

YAZILIM HATA SINIFLANDIRMASINDA FARKLI NAİVE BAYES TEKNİKLERİN KIYASLANMASI

Saadet Aytaç ARPACI (ORCID: 0000-0001-6226-4210)*
Oya KALIPSIZ (ORCID: 0000-0001-9553-669X)

Bilgisayar Mühendisliği Bölümü, Elektrik-Elektronik Fakültesi, Yıldız Teknik Üniversitesi, İstanbul, Türkiye

Geliş / Received: *28.09.2016*
Kabul / Accepted: *22.09.2017*

ÖZ

Bu çalışmada, Mutlak Bağlantılı Ağırlıklandırılmış *Naive Bayes* metodu, *Naive Bayes* metodu ve *Naive Bayes* sınıflandırma metodu üzerine uygulanan çeşitli yumuşatma teknikleri (*Jelinek-Mercer, Dirichlet, Two-Stage*) ile yazılım ölçütlerine dayalı hata sınıflandırmasının karşılaştırmalı analizi araştırılmıştır. Yapılan çalışmada *Chidamber & Kemerer* ve LOC metrik kümesine sahip 3 veri seti üzerinde modellerin başarımı incelenmiştir. Bu çalışma, kullanılan veri setleri/ölçüt gruplarına göre araştırılan metotlardan *Naive Bayes* metodu üzerine uygulanan bazı tekniklerin (*Dirichlet, Two-Stage*) sınıflandırma performansını diğer sınıflandırma metotlarına kıyasla daha da iyileştirdiği sonucunu gösterdi. Bu çalışmanın sonuçlarına göre, %90 üzerinde sınıflandırma doğrulukları DIT-NOC-CBO, DIT-NOC-LCOM, DIT-CBO-LCOM, NOC-CBO-LCOM ölçüt grupları için *Dirichlet* ve *Two-Stage* teknikleriyle elde edildi.

Anahtar Kelimeler: Yazılım hata sınıflandırma, naive bayes, sınıflandırma

A COMPARISON OF DIFFERENT NAIVE BAYES TECHNIQUES FOR SOFTWARE DEFECT CLASSIFICATION

ABSTRACT

In this study was investigated that the comparative analysis of software defect classification with using Absolute Correlation Weighted *Naive Bayes* method, *Naive Bayes* method and various smoothing techniques (*Jelinek-Mercer, Dirichlet, Two-Stage*) on *Naive Bayes* classification technique. In this study, the performance of the models were examined on 3 data sets with a set of metrics *Chidamber & Kemerer* and LOC. The study results showed that according to used data sets/metric groups and methods the performance of some smoothing techniques (*Dirichlet, Two-Stage*) performs better than other classification methods. As the results of this study, over 90% classification accuracies were obtained with *Dirichlet* and *Two-Stage* smoothing techniques on DIT-NOC-CBO, DIT-NOC-LCOM, DIT-CBO-LCOM, NOC-CBO-LCOM metric groups.

Keywords: Software defect classification, naive bayes, classification

*Corresponding author / Sorumlu yazar. Tel.: +90 212 383 57 31; e-mail / e-posta: saadeta99@gmail.com

1. GİRİŞ

Yazılım projelerinde bütçe ve zamanın planlandığı şekliyle ilerlemesi, yazılım gereksinim analizinin doğru oluşturulması ve hatasız bir yazılımın sunulması yazılımın kalitesini yükselten temel faktörlerdir. Yazılım projelerinin zamanla büyümesi ve karmaşıklaşması bakım maliyetlerini de yükseltmekte dolayısıyla yazılımın kalitesi de etkilenmektedir, bu nedenle geliştirilen yazılımlarda oluşan hataların, yazılım geliştirme yaşam döngüsünün erken safhalarında yazılım test faaliyetleri ile tespit edilmesi ve hataların azaltılması ile maliyetlerin de düşmesi sağlanmaktadır [1, 2].

Hataların en aza indirilmesi için test faaliyetlerinin iyileştirilmesine yönelik çok sayıda araştırma yapılmaktadır. Birçok projenin test aşaması içinde toplanan bilgiler analiz edilerek metrikler oluşturulmaktadır, projelerdeki hataların tahmin edilmesinde de bu metriklerden yararlanılmaktadır [1, 3]. Literatürde hatalı olan modüllerin/sınıfların tahmin edilmesi amacıyla makine öğrenme tekniklerini kullanan çok sayıda çalışma bulunmaktadır. Malhotra'nın sunduğu bir incelemede yazılım hata tahmini ve makine öğrenmesi teknikleri üzerine 1991-2013 arası yapılan çalışmaların %47,7'sinin *Bayes* öğrenme tekniğine dayandığı ve bu çalışmaların da %74'ünün *Naive Bayes* (NB) algoritması ile yapıldığı belirtilmektedir. Ayrıca, NB tekniğinin hata tahmin problemleri içinde iyi bir performans sergilediği ve aynı zamanda farklı özellikli çok sayıda veri setini iyi işleyebildiği de rapor edilmektedir [4]. NB'nin basit ve verimli oluşu sınıflandırma alanında halen geniş bir kullanım alanı oluşturmakta ve tekniğin verimliliğini geliştirmeye dayalı çalışmalar da devam etmektedir [4, 5].

NB sınıflandırıcısı bir olasılıksal sınıflandırma yöntemidir. Verinin mevcut sınıflardan herhangi birine ait olma olasılığını hesaplayan bir yaklaşımdır. Bu sınıflandırıcıda nitelikler birbirinden bağımsız olarak kabul edilir. Özelliklerin hepsi aynı derecede öneme sahiptir. Bir özelliğin değeri başka bir özellik değeri hakkında bilgi içermez [6]. NB'nin önemli bir dezavantajı, küçük ölçekli görünmeyen örneklerin olasılık tahminine katılamamasıdır ki bu durumu düzeltmek ve NB verimliliğini yükseltmek için çeşitli yumuşatma teknikleri önerilmiştir [5, 7, 8]. Özellikle metin sınıflandırma alanında bu konu ile ilgili çok sayıda çalışma yapılmıştır. Örneğin, *Jelinek-Mercer* (JM) tekniğinin spam maillerin sınıflandırılması üzerindeki etkisine yönelik bir çalışmada spam mail sınıflandırmasının bu teknik ile daha iyi olduğu ve *False Positive* (FP) değerlerin azaldığı rapor edilmektedir [9]. Metin sınıflandırmaya dayalı diğer çalışmalarda ise, JM, *Dirichlet* (D), *Two-Stage* (TS) ve *Absolute Discount* yumuşatma tekniklerinin NB metodu üzerinde incelenmesi sonucu bu tekniklerin NB metoduna ait doğruluk değerlerini iyileştirdiği belirtilmektedir ve bu çalışmalarda ifade edilen ortak durum ise, en iyi performansı gösteren yumuşatma metodunun TS metodu olduğudur [7, 8, 10]. Metin sınıflandırma haricinde de bu konuya yönelik çalışmalar bulunmaktadır, örneğin NB'e dayalı aktivite sınıflandırmada doğruluk değerlerini iyileştirmek için JM ve D teknikleri önerilmiş ve bu metotların aktivite sınıflandırma doğruluğunda iyileştirme sağladığı gösterilmiş [11], diğer bir çalışmada ise JM metodunun kalp hastalığı tahmininde tahmin başarısını arttırdığı [12] belirtilmektedir.

Özelliklerin aynı derecede öneme sahip olduğunu varsayan NB'e dayalı bu yumuşatma teknikleri haricinde, bazı yazılım hata tahmin çalışmalarında, özelliklerin aynı derecede öneme sahip olamayacağı ve bazı özelliklerin diğerlerine göre daha fazla öneme sahip olduğu bu nedenle de NB tekniğinde her bir özelliğin farklı ağırlık değeri ile yapılandırılması gerektiği belirtilmektedir [13]. Bu doğrultuda NB tekniğine dayalı olarak özelliklerin eşit veya farklı derecede öneme sahip olması durumlarına göre aynı veri seti üzerinde, kullanılan tekniklerin yazılım hata sınıflandırması açısından verimliliğini görebilmek, değerlendirebilmek amacıyla bu araştırmada aşağıda belirtilen araştırma sorularına yanıt bulunmaya çalışılmıştır.

Araştırma soruları:

- Özelliklerin aynı derecede öneme sahip oldukları düşünülürse NB üzerine uygulanan yumuşatma tekniklerinin (*Jelinek-Mercer, Dirichlet, Two-Stage*) doğruluğa etkisi ne olur?
- Özelliklerin farklı derecede öneme sahip oldukları düşünülürse uygulanan Mutlak Bağlantılı Ağırlıklandırılmış *Naive Bayes* tekniğinin doğruluğa etkisi ne olur?
- En yüksek doğruluk değeri hangi teknik ile elde edildi?
- Kullanılan metriklerin doğruluk ölçümünde etkisi var mı?

Çalışmamızda *Chidamber & Kemerer* (CK) metrik (ölçüt) kümesi ve kaynak kod satır sayısı metriğine sahip 3 veri seti üzerinde *Naive Bayes*'e dayalı *Jelinek-Mercer* (JM), *Dirichlet* (D), *Two-Stage* (TS) teknikleri ve Mutlak Bağlantılı Ağırlıklandırılmış *Naive Bayes* (ACWNB [13]) tekniği kullanılarak yazılım hata sınıflandırması için bu tekniklerin başarımı incelenmiştir.

Yazının ikinci bölümünde çalışmada kullanılan metotlar, değerlendirme kriterleri, kaynak kod ölçütleri ile ilgili bilgilere yer verilmiştir. Üçüncü bölümde, çalışmada kullanılan veri seti, ölçüt değerlerinin seçimi ve geliştirilen metot anlatılmaktadır. Dördüncü bölümde, çalışmadan elde edilen bulgular ve bu sonuçlara dayalı değerlendirmeler, beşinci bölümde ise çalışmaya ait genel bir sonuç yer almaktadır.

2. SINIFLANDIRMA MODELİ

Bu bölümde, NB tekniği ve NB tekniğine dayalı çalışmada kullanılan JM, D, TS, ACWNB teknikleri, model başarımının ölçülmesinde kullanılan değerlendirme kriterleri ve çalışmada kullanılan metriklere ait bilgiler sunulmaktadır.

2.1. Sınıflandırma Algoritması

2.1.1. Naive Bayes Sınıflandırma

2.1.1.1. Bayes Teoremi

Olasılık teorisinin iki temel kuralı olan toplam kuralı Eşitlik 1'de, ürün kuralı Eşitlik 2'de belirtilmektedir [14].

$$P(X) = \sum_Y P(X,Y) \tag{1}$$

$$P(X,Y) = P(Y|X)P(X) \tag{2}$$

Eşitliklerde P(X,Y) ortak olasılık, P(Y|X) koşullu olasılık, P(X) marjinal olasılıktır. Ürün kuralına dayanarak, simetri özelliği P(X,Y) = P(Y,X) uygulandığında, Eşitlik 3'te belirtilen *Bayes* kuralının elde edilmesi kolaylaşır [14].

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{3}$$

Toplam kuralının kullanılması ile de *Bayes* teoreminde pay olarak görünen eşitlik payda içinde de ifade edilebilir (Eşitlik 4) [14].

$$P(X) = \sum_Y P(X|Y)P(Y) \tag{4}$$

Verilen veri örneklerinde sınıf olasılıklarının bulunmasında Y değişkeni, veri örnekleri için sınıf etiketini temsil etmektedir ve P(Y=y_j), y_j sınıfı için önsel olasılık olarak ifade edilmektedir. X değişkeni, veri örneklerinin özelliğini temsil etmektedir. Veri setindeki X özelliğine karşın Y sınıfının olma olasılığını hesaplayan P(Y|X) sonsal olasılık değerinin bulunmasında, farklı sınıflar için gözlenen X verisinin olasılığı P(X=x_i|Y=y_j) ve P(Y=yj) önsel olasılık değerlerine ihtiyaç vardır [14].

2.1.1.2. Naive Bayes

Naive Bayes sınıflandırma *Bayes* teoremine dayanır. y_j, j∈ {1, ...,J} olmak üzere J tane sınıftan birini temsil eden ayrık bir değişkendir. X özelliği, m adet özellikten oluşan X=(x_1,x_2,…,x_m) özellik vektörü ile ifade edilmektedir. Olası y_j değeri için *Bayes* teoremine göre sonsal olasılık p(y_j |X) Eşitlik 5'te belirtilmektedir [14].

$$P(y_j|\mathrm{X}) = \frac{P(X|y_j)P(y_j)}{P(X)} \tag{5}$$

Y' hedef sınıfının tahmininde tüm özellikler için koşullu olasılıkların çarpımı hesaplandığında *Naive Bayes* sınıflandırıcı için Eşitlik 6'daki temel eşitlik oluşur [14].

$$Y' \leftarrow argmax_{y_j} \frac{P(y_j)\prod_{i=1}^{m} P(X=x_i|y_j)}{\sum_{j=1}^{j} P(y_j)\prod_{i=1}^{m} P(X=x_i|y_j)} \tag{6}$$

Y' hedef sınıfının tahmini için kullanılan *Naive Bayes* sınıflandırıcıda paydanın tek sınıfa bağımlı olmaması ve tüm hesaplamalarda paydanın ortak olarak işleme katılması nedeniyle Eşitlik 6'daki formül Eşitlik 7'deki gibi sadeleştirilebilir [14].

$$Y' \leftarrow argmax_{y_j} P(Y=y_j)\prod_{i=1}^{m} P(X=x_i|Y=y_j) \tag{7}$$

$$P(y_j) = \frac{|N_j|}{|N|} \tag{8}$$

Eşitlik 8'e göre, sınıf etiketindeki, eğitim niteliklerinin toplam sayısı $|N|$, y_j sınıfındaki eğitim niteliklerinin sayısı $|N_j|$[10].

Her bir sınıf için Eşitlik 7 uygulanıp olasılıklar hesaplandıktan sonra sonuç değerleri içerisinde en büyük olasılığa sahip olan sınıf hedef sınıf olarak seçilir.

2.1.2. Yumuşatma Teknikleri

Yumuşatma ve yumuşatma teknikleriyle ilgili bu bölümde verilen bilgilerde Bai, Nie ve Paradis [10], Patil [12], Aggarwal [15]'ın yazılarından faydalanılmış ve formüller bu çalışmaya uygun olarak düzenlenmiştir.

2.1.2.1. Yumuşatma

Yumuşatma, gürültü nedeniyle veya küçük ölçekli yapılar olması nedeniyle gözden kaçan verideki önemli örneklerin yakalanmasını sağlayan bir tekniktir [12]. Özellik (X) içinde olmayan veri örnekleri NB formülündeki olasılık çarpımlarının sonucunu sıfır yapmaktadır. "Yumuşatma" olarak ifade edilen ve belirli parametre değerlerinin kullanılmasına dayalı olan bazı yöntemlerle bu sonuçlar düzeltilmeye çalışılmaktadır. Bu tekniklerin kullanımı ile sıfır sonucunu veren olasılık değerleri yükseltilirken yüksek olasılık değerlerinin azaltılması sağlanmaktadır böylece yapılan ayarlama sonucunda daha düzgün bir dağılım oluşmaktadır [15]. Yumuşatma metotları içindeki genel formülasyon Eşitlik 9'da belirtilmektedir [10, 15].

$$P(X|y_j) = f(x) = \begin{Bmatrix} P(X|y_j), y_j \textit{ içinde görünen X nitelikleri} \\ P(X|Y),\ y_j \textit{ içinde görünmeyen X nitelikleri} \end{Bmatrix} \tag{9}$$

Böylece y_j sınıfında bulunan örnekler için bir tahmin yapıldığında, y_j sınıfında bulunmayan örnekler için de tahminin tüm veri setine dayandırılmasıyla diğer bir tahmin yapılmış olur. Yumuşatma metodu sadece sıfır olasılığını önlemek ile kalmaz modelin doğruluğunu da iyileştirmeye çalışır [15]. Devam eden satırlarda çalışmada kullanılan yumuşatma metotları incelenmektedir.

2.1.2.2. Jelinek-Mercer Yumuşatma

$$P(X|y_j) = (1-\lambda)P(X|y_j) + \lambda P(X|Y) \tag{10}$$

Tüm veri setine dayalı özelliğin tahmini $P(X|Y)$ ile sınıf modelinin $P(X|y_j)$ maksimum olasılık tahminini linear olarak birleştirir (Eşitlik 10) [10]. λ, (0-1) arasında değişen dengeleme parametresidir [12].

2.1.2.3. Dirichlet Yumuşatma

$$P(X|y_j) = \frac{c(X,y_j)+\mu P(X|Y)}{|y_j|+\mu} \tag{11}$$

Eşitlik 11'de belirtilen formüle göre, $c(X, y_j)$, y_j sınıfı içindeki X örneğinin sayısı, $|y_j|$, y_j sınıfının boyutu [10]. μ, (0-1) arasında değişen bir dengeleme parametresidir [15].

2.1.2.4. Two-stage Yumuşatma

$$P(X|y_j) = (1-\lambda)\frac{c(X,y_j)+\mu P(X|Y)}{|y_j|+\mu} + \lambda P(X|Y) \tag{12}$$

Bu metot, *Dirichlet* ve *Jelinek-Mercer* metotlarının birleştirilmesiyle elde edilir (Eşitlik 12) [10].

2.1.3. Ağırlıklandırılmış Naive Bayes

Naive Bayes metodunun koşullu bağımsızlık özelliği yerine özelliklerin farklı önem derecelerine sahip olduğu düşüncesi uygulanacak olursa bu işlem için önerilen metotlardan biri Asmono, Wahono ve Syukur'un önerdiği

Mutlak Bağlantılı Ağırlıklandırılmış *Naive Bayes* (ACWNB [13]) dir. Mutlak Bağlantılı Ağırlıklandırılmış *Naive Bayes* formülü Eşitlik 13'de belirtilmektedir. w_i, x_i özelliğinin ağırlığı [13].

$$Y' = \arg max_{y_j \in Y} \left(P(y_j) \prod_{i=1}^{m} P(x_i \mid y_j)^{w_i} \right) \quad (13)$$

2.2. Değerlendirme Kriterleri

Doğruluk v.b. birçok performans ölçüm hesaplamalarında modellerin değerlendirilmesi için hata matrisi kullanılmaktadır. Tablo 1'de hata matrisi görülmektedir. Tablo 1'de görülen *True Positive* (TP), *False Positive* (FP), *False Negative* (FN), *True Negative* (TN) ifadeleri aşağıdaki şekilde tanımlanmaktadır [16].

TP: Hatalı örneklerin hatalı olarak tahmini (Hatalı/Hatalı)
FP: Hatasız örneklerin hatalı olarak tahmini (Hatasız/Hatalı)
TN: Hatasız örneklerin hatasız olarak tahmini (Hatasız/Hatasız)
FN: Hatalı örneklerin hatasız olarak tahmini (Hatalı/Hatasız)

Tablo 1. Hata matrisi [16]

Tahmin Edilen Sınıf			
Gerçek Sınıf		**Pozitif**	**Negatif**
	Pozitif	TP	FN
	Negatif	FP	TN

Bu sonuçlarla, hata tahmin literatürlerinde çoğunlukla kullanılan doğruluk değeri elde edilmektedir. Doğruluk, modellerin başarı ölçümünü veren bir yöntemdir. Doğru sınıflandırılmış örnek sayısının (TP +TN), toplam örnek sayısına (TP+TN+FP+FN) oranı olarak ifade edilir [16]. Doğruluk oranı, Eşitlik 14'e göre hesaplanmaktadır.

$$\text{Doğruluk} = (TP + TN) / (TP + FN + FP + TN) \quad (14)$$

2.3. Kaynak Kod Ölçütleri

Malhotra'nın sunduğu incelemede [4], 1991-2013 arası yazılım hata tahmini ile ilgili yapılan çalışmaların bir çoğunda LOC (kaynak kod satır sayısı) gibi boyut ölçümleri ile birlikte *Halstead* ve *Mc Cabe* tarafından tanımlanan geleneksel statik kod metriklerini içeren metriklerin kullanıldığı belirtilmektedir. Bununla birlikte, nesne yönelimli hata metrikleri olarak en fazla LOC veya SLOC içeren çalışmaların görülmüş olduğu ifade edilmektedir. Yazılım hata tahmininde nesne yönelimli metriklerin kullanışlı olduğu ve bu metriklerin içinde de CBO, RFC ve LOC'un en fazla kullanışlı metrikler olduğu, NOC ve DIT metriklerinin ise yazılım hata tahmini için kullanışsız metrikler olduğu belirtilmektedir [4].

2.3.1. Chidamber & Kemerer Metrik Kümesi

Bir kod metriği olan ve literatürde de yaygın olarak kabul görmüş nesneye dayalı yazılım metrik kümeleri içinden CK metrik kümesi 6 temel metrik tanımlamaktadır [17, 18].

1) WMC, bir sınıftaki metotların sayısıdır.
2) DIT, sınıfın, kalıtım ağacındaki kök elemanına uzaklığıdır.
3) NOC, bir sınıftan türetilmiş alt sınıfların sayısıdır.
4) CBO, bir sınıfın, aralarında kalıtım olmayan diğer sınıfların özellik ve metotlarını kullanarak bağlandığı farklı sınıfların sayısıdır.
5) RFC, bir sınıftan bir nesnenin metotları çağrıldığında, bu nesnenin tetikleyebileceği tüm metotların sayısıdır.
6) LCOM, aynı özelliklerden birine veya daha fazlasına erişen metotların sayısıdır.

3. MATERYAL VE METOT

Bu bölüm içinde, gerçekleştirilen deneysel çalışmada kullanılan veri seti, metrikler ve geliştirilen yöntem hakkında bilgi verilmiştir.

3.1. Veri Seti

Yazılım hata sınıflandırması açısından modelleri inceleyebilmek amacıyla bu çalışmada [19] adresinden elde edilen *Marco D'Ambros* ve arkadaşları tarafından hazırlanmış veri setleri kullanılmıştır. Bu veri setleri içinde *Eclipse JDT* yazılımında 997, *Eclipse PDE UI* yazılımında 1497, *Eclipse Equinox* çerçevesinde 324 örnek incelenmiştir. Veri seti içerisinde, CK, *Object Oriented* gibi örneklere ait farklı metrikler ve örneklerin hata sayıları bulunmaktadır. Çalışmamızda kullanılan ölçütlerin seçimi "Ölçüt Değerlerinin Seçimi" bölümünde anlatılmaktadır. Veri setlerine ait önemli (major) hata bulunma oranları Tablo 2'de belirtilmektedir.

Tablo 2. Veri setlerinde önemli (major) hata bulunma yüzdesel oranları

	Hata 0 (%)	**Hata 1 (%)**	**Hata 2 (%)**	**Hata 3 (%)**
Eclipse JDT	97,09	2,41	0,40	0,10
Eclipse PDE UI	96,53	3,14	0,33	-
Equinox Framework	98,76	1,24	-	-

Tablo 3. Çalışmada kullanılan metrik grupları

GRUP NO	**GRUP**	**GRUP NO**	**GRUP**
1	WMC-DIT-NOC	18	DIT-NOC-LCOM
2	WMC-DIT-CBO	19	DIT-NOC-LOC
3	WMC-DIT-RFC	20	DIT-CBO-RFC
4	WMC-DIT-LCOM	21	DIT-CBO-LCOM
5	WMC-DIT-LOC	22	DIT-CBO-LOC
6	WMC-NOC-CBO	23	DIT-RFC-LCOM
7	WMC-NOC-RFC	24	DIT-RFC-LOC
8	WMC-NOC-LCOM	25	DIT-LCOM-LOC
9	WMC-NOC-LOC	26	NOC-CBO-RFC
10	WMC-CBO-RFC	27	NOC-CBO-LCOM
11	WMC-CBO-LCOM	28	NOC-CBO-LOC
12	WMC-CBO-LOC	29	NOC-RFC-LCOM
13	WMC-RFC-LCOM	30	NOC-RFC-LOC
14	WMC-RFC-LOC	31	CBO-RFC-LCOM
15	WMC-LCOM-LOC	32	CBO-RFC-LOC
16	DIT-NOC-CBO	33	RFC-LCOM-LOC
17	DIT-NOC-RFC	34	NOC-LCOM-LOC

3.2. Ölçüt Değerlerinin Seçimi

Yazılım metriklerinin özellikleri hata tahmin modelinin verimlilik ve performansını etkilemektedir. Bazı metriklerin gereksiz bilgi sunması da hata tahmin modeli üzerinde zıt etki oluşturmaktadır. Bu nedenle yazılım metriklerinin tümünün kullanılması yerine, seçimi yazılım hata tahmini için önemlidir ve uygulayıcılar için de yararlıdır. Bu doğrultuda verilen sistem için etkili bir yazılım hata tahmin modelinin kurulmasında kullanılabilecek yazılım metrik sayısı [20] makalesine göre üç olarak belirtilmektedir [20].

Verimli bir yazılım hata tahmini için belirtilen bu metrik sayısına ilaveten hangi metriklerin kullanılmasının yazılım hata tahmininde daha belirleyici olabileceğine dair yapılmış bir çalışmada NR, NDC, NML, NDPV, LOC, RFC, CBO, AMC metrikleri yazılım hata tahmininde gösterdikleri ayırıcılık özellikleri nedeniyle öncelikli olarak tavsiye edilmektedir. Bununla birlikte yazılım projeleri ve hata belirteçleri arasındaki farklar nedeniyle bir projede iyi çalışan belirtecin diğerinde işe yaramayabileceği de belirtilmektedir [21].

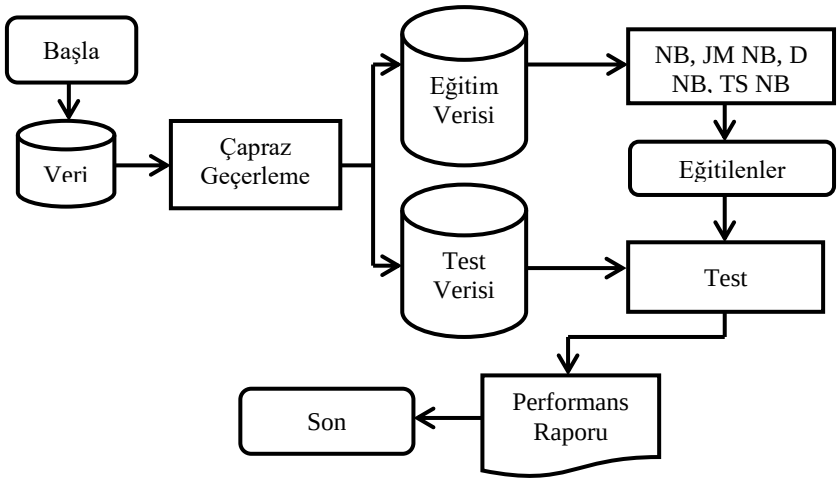
Çalışmamızda, CK metriklerinden ve genel olarak yazılım hata tahmin çalışmalarında çok kullanılan bir metrik olması nedeniyle LOC metriğinden yararlanıldı. Çalışmamızdaki metotların hangi metrikler ile daha verimli sonuç alacağını anlayabilmek amacıyla LOC ve CK metriklerinden oluşan her biri üç metriğe sahip 34 ayrı metrik grubu üzerinde metotların başarımı incelendi. Tablo 3'de çalışmada kullanılan gruplar belirtilmektedir.

3.3. Geliştirilen Metot

Çalışmamızda yazılım hata sınıflandırması için kullanılan *Naive Bayes* tekniğine dayalı metotlar *Eclipse* platformunda java programlama dili kullanılarak geliştirildi. Özelliklerin aynı derecede öneme sahip olması ve olmaması durumlarına göre çalışma iki farklı yol ile devam eden satırlarda belirtildiği gibi oluşturuldu.

3.3.1. Özelliklerin Aynı Derecede Öneme Sahip Olması Durumu

Yazılım hata sınıflandırmasına yönelik kullanılan modellerin başarımını belirlemek için kurulan çerçeve Şekil 1'de belirtilmektedir. Veri seti 5 kat çapraz geçerleme işlemi ile eğitim ve test verisi olarak ayrıldı. Oluşan eğitim ve test kümelerinden, eğitim seti üzerinde *Naive Bayes* algoritması ve *Naive Bayes* tekniğine dayalı JM, D, TS metotlarının her birinin ayrı şekilde eğitiminin sağlanıp test kümesi ile de eğitilenlerin doğruluklarının değerlendirilmesi yapıldı. Değerlendirme en yüksek olasılığa sahip sınıfa göre yapıldı. Elde edilen sonuçlar dördüncü bölümde belirtilmektedir.



Şekil 1. Özelliklerin aynı derecede öneme sahip olması durumuna göre kurulmuş çerçeve

3.3.2. Özelliklerin Farklı Derecelerde Öneme Sahip Olması Durumu

Yazılım hata sınıflandırmasına yönelik kullanılan modelin başarımını belirlemek için kurulan çerçeve Şekil 2'de belirtilmektedir. Veri seti 5 kat çapraz geçerleme işlemi ile eğitim ve test verisi olarak ayrıldı. Eğitim verileri eğitim sürecinde kullanıldı. Eğitim işleminde, ortalamalar (μ), standart sapma (σ) değerleri ve her bir özelliğin ağırlığı (w_i) devam eden satırlarda belirtildiği şekliyle hesaplandı.

1) Her bir sınıf içindeki özelliğin ortalama değeri hesaplandı. Özelliğin tüm örnek değerlerinin toplanması ve örnek miktarına bölünmesi ile ortalama değer elde edildi (Eşitlik 15) [13].

$$\mu = \frac{a_1+a_2+\cdots+a_n}{n} \tag{15}$$

2) Her bir sınıf içindeki özelliğin standart sapması hesaplandı. Standart sapma (σ) varyansın (σ^2) karekök değeridir. Her bir sınıfa ait özelliklerin örnek değerlerinden o sınıfın özelliğine ait ortalama değerin (μ) çıkarılıp karelerinin alınması ve sonuçta örnek miktarının bir eksiğine bölünmesiyle elde edildi (Eşitlik 16) [13].

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(a_i-\mu)^2 \tag{16}$$

3) Ağırlığın hesaplanmasında, sınıf ortalaması ile diğer sınıfların ortalamaları arasındaki farkın standart sapma değerlerinin toplamına bölünmesi ve sonucun mutlak değer ile elde edilmesi sağlanmıştır (Eşitlik 17) [13].

$$w_i = \left|\frac{(\mu_{ij}-\mu_{\bar{i}j})}{(\sigma_{ij}+\sigma_{\bar{i}j})}\right| \tag{17}$$

w_i, i özelliğinin ağırlığı, μ_{ij}, j sınıfı için i özelliğinin ortalama değeri, $\mu_{i\bar{j}}$, j dışındaki sınıflar için i özelliğinin ortalama değeri, σ_{ij}, j sınıfı için i özelliğinin değerlerinin standart sapması ve $\sigma_{i\bar{j}}$, j dışındaki sınıflar için i özelliğinin değerlerinin standart sapması [13].

Eğitim sürecinde geliştirilen model, kullanılan test verileri ile test edildi. Test sürecinde, ağırlıklandırılmış olasılık ve önsel olasılık hesaplandı.

1) Ağırlıklandırılmış olasılık tayininde Eşitlik 18'de belirtilen normal dağılım kullanıldı. Burada eğitim sürecinde elde edilen ortalama ve standart sapma değerleri kullanıldı [13].

$$P(x|y) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{(a-\mu)^2}{2\sigma^2}} \tag{18}$$

Her bir özelliğin normal dağılımı, eğitim sürecinde elde edilen ağırlık değerleriyle üsleri alınarak hesaplandı [13]. Ağırlıklandırılmış olasılık formülü Eşitlik 19'da belirtilmektedir.

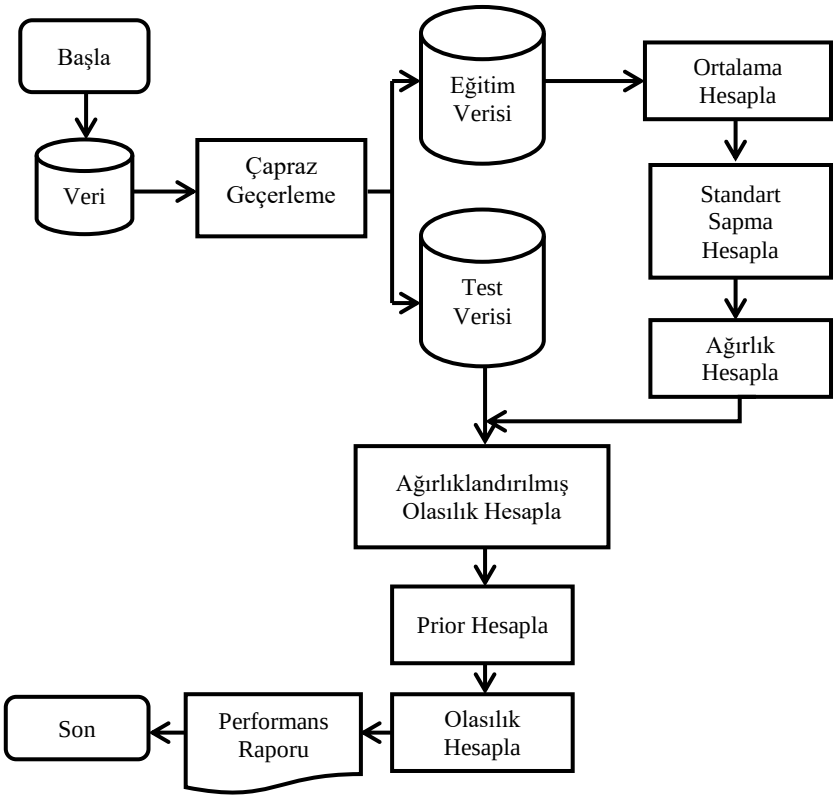
$$L = \prod_{i=1}^{m} p(x_i \mid y_j)^{w_i} \tag{19}$$

2) Önsel olasılık (*prior*) hesaplandı. Sınıfın ağırlıklandırılmış olasılık değerinin tüm sınıfların ağırlıklandırılmış olasılık değerlerinin toplamına bölümü ile elde edildi [13]. Formül, Eşitlik 20'de belirtilmektedir.

$$p(y_j) = \frac{\prod_{i=1}^{m} p(x_i \mid y_j)^{w_i}}{\prod_{i=1}^{m} p(x_i \mid y_j)^{w_i} + \prod_{i=1}^{m} p(x_i \mid \overline{y_j})^{w_i}} \tag{20}$$

3) Sınıf olasılığı hesaplandı. Ağırlıklandırılmış olasılık değerinin önsel olasılık ile çarpımı sonucu sınıf olasılığı bulundu [13]. Eşitlik 21'de sınıf olasılığı formüle edilmektedir.

$$Y' = p(y_j) \prod_{i=1}^{m} p(x_i \mid y_j)^{w_i} \tag{21}$$



Şekil 2. Özelliklerin farklı derecede öneme sahip olması durumuna göre kurulmuş çerçeve [13]

Tüm sınıflar için 1-3 basamaklarında belirtilen işlemler tekrarlandı. En yüksek olasılığa sahip sınıfa göre değerlendirme yapıldı. Elde edilen sonuç dördüncü bölümde belirtilmektedir.

4. BULGULAR VE TARTIŞMA

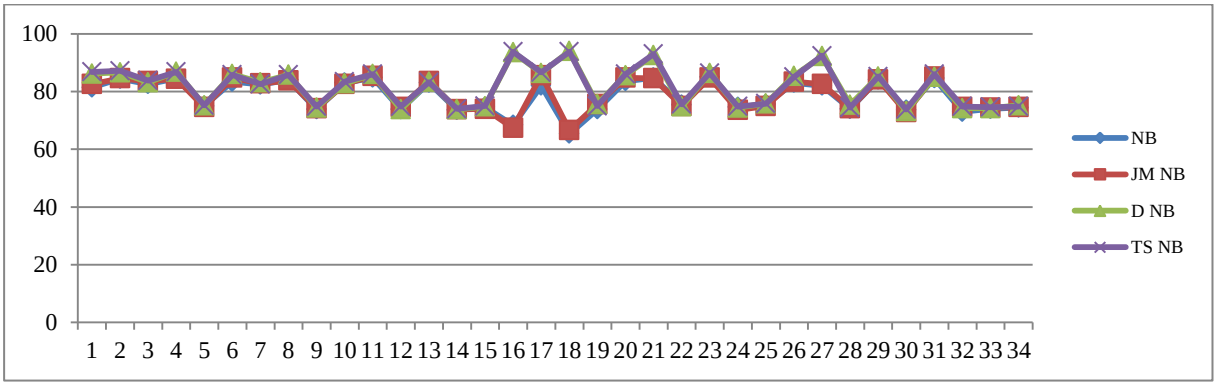
Bu çalışmada yazılım metriklerine yönelik 3 veri seti üzerinde NB, JM NB, D NB, TS NB ve ACWNB teknikleri denenmiştir. Özelliklerin aynı veya farklı derecede öneme sahip olması durumuna göre incelenen verilere ait sonuçlar devam eden satırlarda sunulmaktadır.

4.1. Özelliklerin Aynı Derecede Öneme Sahip Olması Durumuna Göre Elde Edilen Sonuç

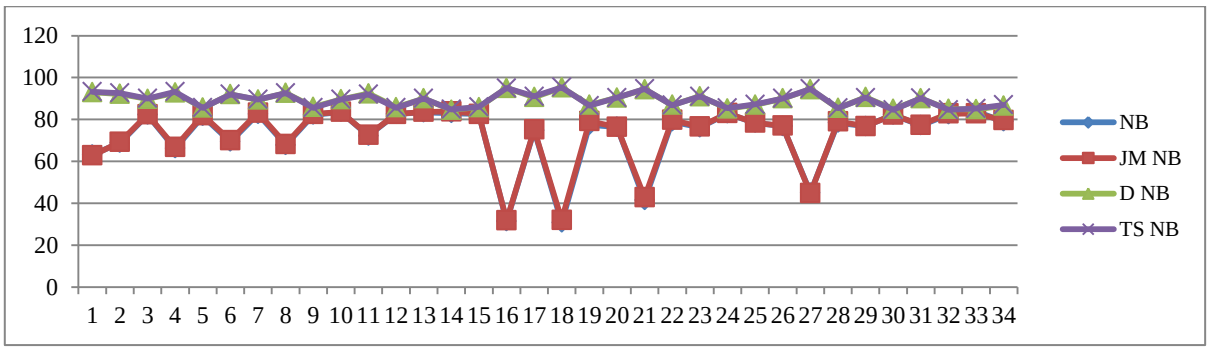
Eclipse JDT, PDE UI, Equinox Framework veri setleri üzerinde μ=0,2, λ=0,9 değerlerine göre 34 metrik grubu için elde edilen doğruluk sonuçları devam eden satırlarda belirtilmektedir.

Şekil 3'te görüldüğü gibi *Eclipse JDT Core* veri setinde genel olarak D NB ve TS NB metotlarının ortalama doğruluk yüzdeleri tüm gruplarda yakın değerler almıştır, JM NB ve NB metotları da tüm gruplar için yakın doğruluk değerlerine ulaşmıştır. D NB ve TS NB metotlarının 16, 18, 21 ve 27. gruplar için doğruluk yüzdelerinin NB ve JM NB metotlarına göre daha fazla olduğu, %90'ı aştığı gözlendi (Şekil 3).

Eclipse PDE UI veri setinde D NB ve TS NB metotlarının ortalama doğruluk yüzdelerinin birbirlerine yakın olduğu görüldü. Aynı şekilde NB metodu ile JM NB metodunun ortalama doğruluk değerlerinin yakınlığı gözlendi. Tüm metotların ortalama doğruluk değerlerini gösteren Şekil 4'teki grafiğe göre, tüm gruplar için D NB ve TS NB metotlarının NB ve JM NB metotlarına göre daha belirgin ve yüksek ortalama doğruluk değerlerine ulaştığı gözlendi. Ayrıca, bu veri setinde de D NB ve TS NB metotları ile JM NB ve NB metotları arasında ortalama doğruluk değerlerinde 16, 18, 21, 27. gruplar için çok daha yüksek farklar oluştuğu gözlendi (Şekil 4).



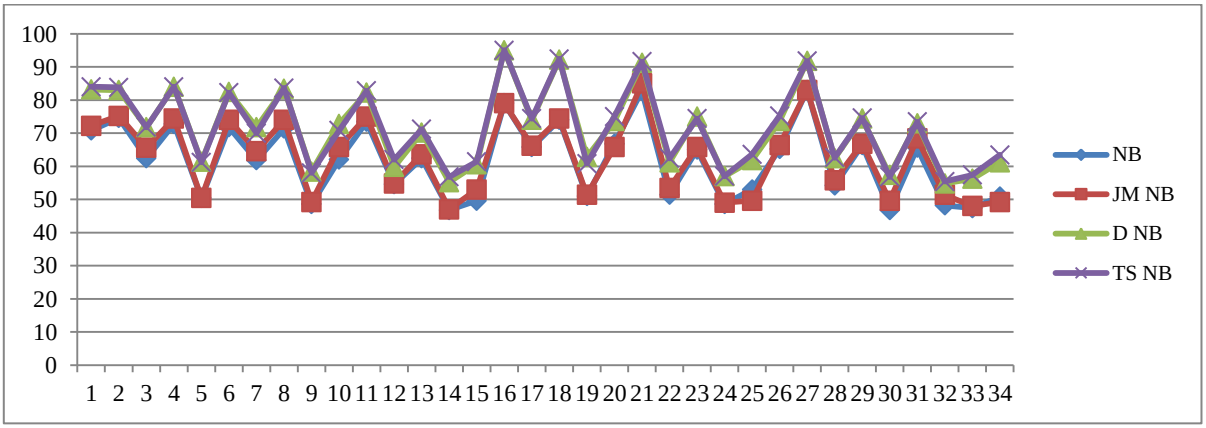
Şekil 3. *Eclipse JDT Core* veri setinde NB, JM NB, D NB, TS NB metotlarının tüm gruplar için doğruluk değerlerini gösteren grafik



Şekil 4. *Eclipse PDE UI* veri setinde NB, JM NB, D NB, TS NB metotlarının tüm gruplar için doğruluk değerlerini gösteren grafik

Equinox Framework veri setinde de NB ve JM NB metotları ortalama doğruluk yüzdeleri açısından birbirlerine yakın değerler alırken D NB ve TS NB metotlarının da birbirlerine yakın değerler aldıkları gözlendi (Şekil 5). D NB ve TS NB metotlarının ortalama doğruluk değerlerinin daha yüksek olduğu ve NB, JM NB metotlarıyla belirgin bir ayrımın olduğu Şekil 5'teki grafikte de görülmektedir. 16, 18, 21, 27. gruplarda bu veri seti için de en yüksek değerlerin D NB ve TS NB metotları ile alındığı görüldü (Şekil 5).

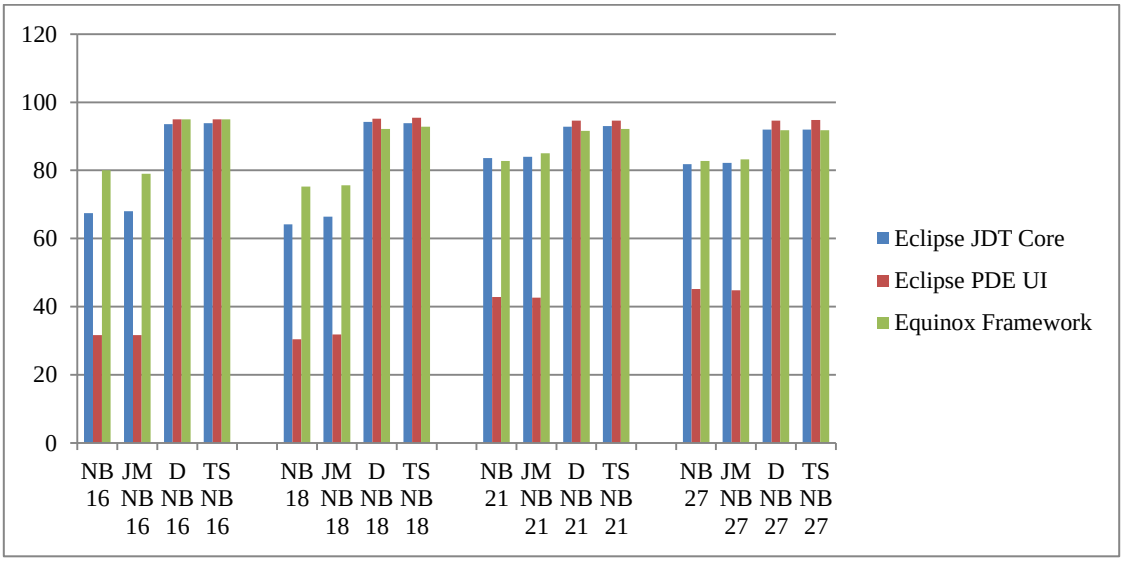
Üç veri setine ait ortalama doğruluk değerlerini ifade eden bu sonuç bilgilerindeki en belirgin durum, 16, 18, 21, 27. gruplar için D NB ve TS NB metotlarının yüksek doğruluk değerlerine ulaşmış olmasıdır. Veri setlerindeki 16, 18, 21, 27. grupları için NB, JM NB, D NB, TS NB metotları ile gözlenen doğruluk değerleri Tablo 4'de sunulmaktadır. Tablonun grafiksel gösterimi de Şekil 6'da görülmektedir.



Şekil 5. *Equinox Framework* veri setinde NB, JM NB, D NB, TS NB metotlarının tüm gruplar için doğruluk değerlerini gösteren grafik

Tablo 4. Tüm veri setlerinde uygulanan NB, JM NB, D NB, TS NB metotlarının 16, 18, 21, 27. gruplarda gösterdiği doğruluk değerleri

	Eclipse JDT Core	**Eclipse PDE UI**	**Equinox Framework**
NB 16	67,4	31,6	80,0
JM NB 16	68,0	31,6	79,0
D NB 16	93,6	95,0	95,0
TS NB 16	93,8	95,0	95,0
NB 18	64,2	30,4	75,2
JM NB 18	66,4	31,8	75,6
D NB 18	94,2	95,2	92,2
TS NB 18	93,8	95,4	92,8
NB 21	83,6	42,8	82,8
JM NB 21	84,0	42,6	85,0
D NB 21	92,8	94,6	91,6
TS NB 21	93,0	94,6	92,2
NB 27	81,8	45,2	82,8
JM NB 27	82,2	44,8	83,2
D NB 27	92,0	94,6	91,8
TS NB 27	92,0	94,8	91,8



Şekil 6. Tüm veri setlerinde, uygulanan NB, JM NB, D NB, TS NB metotlarının 16, 18, 21, 27. gruplarda gösterdiği doğruluk değerleri

Çalışmamızda TS NB metodunun [7, 8, 10] çalışmalarında da belirtilen yüksek doğruluk değerlerine ulaştığı görülmesine rağmen [9, 12] çalışmalarında ifade edilen JM NB'nin sınıflandırma başarısını yükseltmesine dair belirgin bir durum gözlenmemiştir. Çalışmamızda, JM NB yazılım hata sınıflandırma başarısı açısından öne çıkamamıştır.

NB, JM NB, D NB, TS NB tekniklerinin doğruluk değerleri [21] makalesinde tavsiye edilen metrikleri içeren 32. grup için yüksek farklılık göstermemiştir. Bu tekniklerden D NB ve TS NB tekniklerinin, her 3 veri seti için de yüksek doğruluk değerleri göstermesine neden olan metrik grupları 16, 18, 21 ve 27. gruplar olmuştur.

Yazılım hata tahmini için CBO, RFC ve LOC'un en fazla kullanışlı metrikler [4, 21], NOC ve DIT metriklerinin ise kullanışsız metrikler olarak belirtilmesine [4] rağmen araştırmalarda kullanılan metot farklılıkları nedeniyle farklı sonuçlara ulaşılması da bir gerçektir. Çalışmamızda 16, 18, 21 ve 27. gruplarda D NB ve TS NB metotları için yüksek bir doğruluk değeri elde edilmiştir. 16, 18, 21 ve 27. grupların metrik içerikleri grup numaralarına göre sırasıyla, DIT-NOC-CBO, DIT-NOC-LCOM, DIT-CBO-LCOM, NOC-CBO-LCOM dur. Bu metrik gruplarında dikkati çeken durum ise hepsinde DIT ve/veya NOC metriklerinin bulunmasıdır. Bu doğrultuda DIT ve/veya NOC metriklerinin CBO ve/veya LCOM metrikleriyle birlikte D NB ve TS NB metotları için verimli sonuçlar oluşturduğu söylenebilir.

D NB ve TS NB metotlarının 16, 18, 21 ve 27 gruplarında NB ve JM NB metotlarına kıyasla gösterdiği yüksek doğruluk değerleri hata matrisi elemanlarından (TP, TN, FP, FN) hangisinin değerinin yükseltilmesi veya azaltılması nedeniyle elde edilmiştir? Yapılan gözlemlere göre 16, 18, 21 ve 27. gruplarda TN (Hatasız örneklerin hatasız olarak tahmini) değerlerinin artması, TP (Hatalı örneklerin hatalı olarak tahmini) değerlerinin sıfır ya da çok düşük değerleri alması, FP (Hatasız örneklerin hatalı olarak tahmini) değerlerinin sıfır olması (oluşmaması) ve FN (Hatalı örneklerin hatasız olarak tahmini) değerlerinin düşmesi D NB ve TS NB metotlarının bu gruplar için yüksek doğruluk değerlerine ulaşmasına neden olmuştur.

Veri setindeki hatalı ve hatasız veri oranlarıyla paralel bir uyumluluğu en iyi gösteren D NB ve TS NB metotlarının FP değerlerini göstermemesi ve FN değerlerini düşürmesi bu tekniklerin kullanılan veri setleri/metrik grupları için tahmin başarısında oluşturduğu bir avantaj olarak görülebilir. Gösterdikleri tahmin başarımı nedeniyle, yazılım hata tahmin çalışmaları açısından literatürde kullanılmak üzere D NB ve TS NB metotları önerilebilir.

Çalışmalarda, sınıflandırma performansının, yumuşatma metotlarının seçimine [10] ve onların parametrelerine [9, 10] yüksek derecede bağımlı olduğu ifade edilmektedir. Yumuşatma tekniklerine ait parametre (μ, λ) değerlerinin doğru belirlenmesi halinde sınıflandırmanın verimli olabileceği belirtilmektedir [9, 10]. Buna göre, çalışmamızda da kullanılan tekniklerin doğruluklarını yükseltmek amacıyla parametre değerleri değiştirilerek bazı sonuçlara ulaşılmıştır.

Çalışmamızda D NB ve TS NB metotları *Eclipse PDE UI* veri setinin 18. grubu ile en yüksek doğruluk değerine ulaşmıştır. *Eclipse PDE UI* veri setinin 18. grubu için D NB metodunun μ değeri 0,5, TS NB metodunun μ=0,3 λ=0,6, μ=0,2 λ=0,4, μ=0,6 λ=0,3 değerleri alındığında en yüksek ortalama doğruluk değerinin %95,6'ya ulaştığı gözlendi.

Bu çalışmada da uygun parametre değerleri verildiğinde tekniklerin doğruluk değerleri üzerindeki etkisi hafif de olsa bir artış olarak kendini göstermektedir.

4.2. Özelliklerin Farklı Derecelerde Öneme Sahip Olması Durumuna Göre Elde Edilen Sonuç

Eclipse JDT, PDE UI, Equinox Framework veri setleri üzerinde Mutlak Bağlantılı Ağırlıklandırılmış *Naive Bayes* (ACWNB) tekniğinin 34 metrik grubu için elde edilen doğruluk sonuçları incelendiğinde ACWNB tekniği için metrik grupları arasında aldıkları doğruluk değerleri açısından farklar görülmüştür. Tüm metrik grupları dikkate alındığında ACWNB tekniği ile ulaşılan doğruluk değerlerinin %60 üzerine çıkamadığı ve NB, JM NB, D NB, TS NB teknikleri ile elde edilen doğruluk değerlerine oranla aldıkları değerlerin daha düşük olduğu görülmüştür. Bu durum grup 4 örneği ile de Tablo 5'de örneklendirilmektedir. Bununla birlikte tüm veri setlerinde ACWNB metodu ile grup 34'ün (0-1,2) aralığında doğruluk değeri aldığı da gözlenmiştir. Aynı aralıktaki değerler 8, 13, 14, 15, 29, 33 gruplarında da gözlenmiştir. Bu sonuçlar dahilinde ACWNB tekniğinin kullanılan veri setleri ve/veya oluşturulan metrik gruplarıyla uyumlu olmadığı söylenebilir.

Çalışmamızda kullanılan veri setleri ve metrik gruplarına göre NB için özelliklerin eşit derecede öneme sahip olduğu düşünüldüğünde uygulanan tekniklerin, ACWNB tekniğine göre doğruluk değerlerinin daha verimli olduğu görülmüştür.

Tablo 5. Tüm veri setlerindeki 4. gruplarda metotların elde ettikleri doğruluk değerleri

Veri seti/Metotlar	NB	JM NB	D NB	TS NB	ACWNB
Eclipse JDT Core	84,6	84,4	86,8	86,8	30,4
Eclipse PDE UI	66,2	66,8	92,8	93,2	43,8
Equinox Framework	72,8	74,4	84,0	84,0	37,6

5. SONUÇLAR

Bu çalışmada, *Naive Bayes* ile birlikte literatürde *Naive Bayes* metodunun doğruluk değerini iyileştirdiği ifade edilen *Jelinek-Mercer, Dirichlet, Two-Stage* teknikleri ve Mutlak Bağlantılı Ağırlıklandırılmış *Naive Bayes* tekniğinden oluşan 5 ayrı metot ile LOC ve CK metrik kümesine dayalı oluşturulan 3 veri setindeki 34 ayrı ölçüt grubunun çalışma uyumlulukları değerlendirilip, sınıflandırma başarıları kıyaslanmıştır.

Deney sonuçlarında *Naive Bayes* ve *Jelinek-Mercer, Dirichlet, Two-Stage* yumuşatma metotlarıyla birlikte kullanılan *Naive Bayes* metotlarının Mutlak Bağlantılı Ağırlıklandırılmış *Naive Bayes* tekniğine göre daha yüksek doğruluk değerlerine ulaştığı gözlenmiştir. Yüksek doğruluk değerlerine ulaşan bu teknikler arasında ise %90 üzerinde bir sınıflandırma doğruluğu ile en iyi performansı *Dirichlet* ve *Two-Stage* metotlarının veri setlerindeki DIT ve/veya NOC metriklerinin CBO ve/veya LCOM metrikleriyle birlikte oluşturduğu üçlü gruplar ile gösterdiği gözlenmiştir. Bu metrik gruplarında *Dirichlet* ve *Two-Stage* metotlarının hata matrisi elemanlarından TN değerlerinin yükseltilmesi, TP değerlerinde sıfır ya da çok düşük değerler alınması, FP değerlerinin sıfır olması ve FN değerlerinin düşürülmesi ile yüksek doğruluk değerlerine ulaştıkları gözlenmiştir. Yüksek doğruluk değerleri gösteren *Dirichlet* ve *Two-Stage* metotlarının daha iyi bir performans sergileyebilmeleri hedefiyle parametre değerleri de değiştirilmiştir. Buna göre *Eclipse PDE UI* veri setinin DIT-NOC-LCOM ölçüt grubu ile en yüksek doğruluk değerine ulaştıkları gözlenmiştir. Bu metotlardan *Dirichlet* metodunun μ değerinin 0,5, *Two-Stage* metodunun μ=0,3 λ=0,6, μ=0,2 λ=0,4, μ=0,6 λ=0,3 değerleri için en yüksek ortalama doğruluk değeri olan %95.6 ya ulaştığı gözlenmiştir.

Bu çalışmanın sonucunda yazılım hata sınıflandırması çerçevesinde, uygulayıcılara yüksek doğruluk değerlerine ulaşan metotlar ve metotların uyumlu olarak çalışabildikleri az sayıdaki metrik gruplarına ait bir özet sunulmaktadır.

KAYNAKLAR

[1] ERTEMEL, H.Ö., SELÇUK, Y.E., KALIPSIZ, O., "A Cohesion Metric Proposal for Object-Oriented Systems: Comias", 13th WSEAS Int'l. Conf. on Computers, 575-580. Rhodes, Greece, 2009.

[2] BOEHM, B., BASILI, V.R., "Software Defect Reduction Top 10 List", IEEE Computer, 34, 135-137, 2001.
[3] https://www.ijcsi.org/papers/IJCSI-9-5-2-288-296.pdf (erişim tarihi 19.12.2017)
[4] MALHOTRA, R., "A Systematic Review of Machine Learning Techniques for Software Fault Prediction", Applied Soft Computing, 27, 504–518, 2015.
[5] KAUR, G., OBERAI, E.N., "A Review Article on Naive Bayes Classifier with Various Smoothing Techniques", International Journal of Computer Science and Mobile Computing, 3, 864-868, 2014.
[6] KARAKOYUN, M., HACIBEYOĞLU, M., "Biyomedikal Veri Kümeleri ile Makine Öğrenmesi Sınıflandırma Algoritmalarının İstatistiksel Olarak Karşılaştırılması", D.E.Ü. Mühendislik Fakültesi Mühendislik Bilimleri Dergisi, 16, 30-41, 2014.
[7] YUAN, Q., CONG, G., THALMANN, N.M., "Enhancing Naive Bayes with Various Smoothing Methods for Short Text Classification", Proceedings of the 21st International Conference on World Wide Web, 645-646. Lyon, France, 2012.
[8] AGGARWAL, S., "Naive Bayes Classifier with Various Smoothing Techniques for Text Documents", International Journal of Computer Trends and Technology, 4, 873-876, 2013.
[9] ADEWOLE, A.P., FAKOREDE, O.J., AKWUEGBO, S.O.N., " Evaluation of Linear Interpolation Smoothing on Naive Bayes Spam Classifier", International Journal of Technology Enhancements and Emerging Engineering Research, 2, 143-146, 2014.
[10] http://rali.iro.umontreal.ca/rali/sites/default/files/publis/LMforTextClassification.pdf (erişim tarihi 19.12.2017)
[11] SARKAR, A.M.J., LEE, Y.K., LEE, S., "A Smoothed Naive Bayes-Based Classifier for Activity Recognition", IETE Technical Review, 27, 107–119, 2010.
[12] PATIL, R.R., "Heart Disease Prediction System Using Naive Bayes and Jelinek-Mercer Smoothing", International Journal of Advanced Research in Computer and Communication Engineering, 3, 6787-6792, 2014.
[13] ASMONO, R.T., WAHONO, R.S., SYUKUR, A., "Absolute Correlation Weighted Naive Bayes for Software Defect Prediction", Journal of Software Engineering, 1, 38-45, 2015.
[14] DENG, H., SUN, Y., CHANG, Y., HAN, J., Probabilistic Models for Classification. C.C. AGGARWAL (Eds.), Data Classification Algorithms and Applications (pp. 67-70), CRC Press, New York, USA, 2015.
[15] AGGARWAL, S., "Enhanced Smoothing Methods Using Naïve Bayes Classifier for Better Spam Classification", International Journal of Engineering Research and Technology, 2, 3061-3073, 2013.
[16] http://research.ijcaonline.org/icccmit2014/number2/icccmit7017.pdf (erişim tarihi 19.12.2017)
[17] PRESSMAN, R.S., Software Engineering a Practitioner's Approach (7th ed.), McGraw-Hill, New York, USA, 2010.
[18] http://dspace.yildiz.edu.tr:8080/xmlui/bitstream/handle/20.500.11871/170/0036786.pdf?sequence=1&isAllowed=y (erişim tarihi 19.12.2017)
[19] http://bug.inf.usi.ch/ (erişim tarihi 11.06.2016)
[20] WANG, H., KHOSHGOFTAAR, T. M., SELIYA, N., "How Many Software Metrics Should be Selected for Defect Prediction?", In Proceedings of the 24th Florida Artificial Intelligence Research Society Conference, 69-74. Florida, USA, 2011.
[21] JURECZKO, M., "Significance of Different Software Metrics in Defect Prediction", Software Engineering: An International Journal, 1, 86-95, 2011.