Euroasia Journal of Mathematics, Engineering, Natural & Medical Sciences
International Indexed and Refereed
ISSN 2667-6702

Weka Öznitelik Seçim Metotları ile Makine Öğrenmesi Algoritmalarının Performanslarının Karşılaştırılması

Comparison of the Performances of Machine Learning Algorithms Using WEKA Feature Selection Methods

Zeynep Behrin Güven Aydın[1], **Rüya Şamlı**[2]

[1]Dr. Öğr. Üyesi, Doğuş Üniversitesi, Mühendislik Fakültesi, Yazılım Mühendisliği Bölümü, İstanbul/Türkiye

[2]Prof. Dr., İstanbul Üniversitesi-Cerrahpaşa, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, İstanbul/Türkiye

** Corresponding author: zguven@dogus.edu.tr*

Geliş Tarihi / Received: 12.10.2024
Kabul Tarihi / Accepted: 23.11.2024

Araştırma Makalesi/Research Article
DOI: 10.5281/zenodo.14568594

ÖZET

Bu çalışmada yazılım hata tahmini konusunda literatürde yayınlanmış birçok yayında yer alan öznitelik seçimi konusu araştırılmıştır. Öznitelik seçimi, genellikle veri setlerindeki ilgisiz ve gereksiz öznitelikleri azaltarak sınıflandırıcının doğruluğunu arttırmak amaçlı kullanılır Çalışmada NASA veri setleri ve deneysel veri seti üzerinde farklı özellik çıkarım metotları denenerek, seçilen en uygun iki tanesi olan Cfs Subset Eval algoritması ve Temel Bileşen öznitelik seçim metotları ile işlemler gerçekleştirilmiştir. Bunun sonucu olarak hangi algoritmaların başarı oranlarının daha yüksek olduğu tespit edilmeye çalışılmıştır. Elde edilen sonuçlar incelendiğinde, genellikle doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark olduğu görülmektedir.

JM1, KC1, CM1 ve PC1 veri setleri üzerinde farklı öznitelik çıkarım metotları test edildiğinde, tüm veri setlerinde yer alan 22 öznitelik, en uygun metot olan Cfs Subset Eval algoritması ve Principal Component öznitelik seçim metotlarının seçilmesiyle birlikte 8 özniteliğe düşürülmüştür. Daha sonra WEKA platformunda 46 adet sınıflandırma algoritmasının doğruluk oranları hesaplanmıştır. Tüm veri setlerinde doğruluk oranlarında en iyi değişim Bayes Net, Voted Perceptron, K* ve Random Forest algoritmalarında görülmüştür. NASA veri setleri ve deneysel veri setleri üzerinde uygulanan tüm öznitelik seçim metotlarında yazılım metriklerine ait loc, n, v ve defect özniteliklerinin kesinlikle olması gerektiği görülmüştür. Her veri setini oluşturan yazılım metriklerinin hesaplamasında loc (kod satır sayısı), n(tekil operatör ve tekil operand sayısı toplamı), v(program hacmi) ve defect (hata olup olmadığı) özniteliklerinin oldukça önemli olduğu açıkça belli olmuştur.

Anahtar Kelimeler: Hata tahmini, makine öğrenmesi algoritmaları, öz nitelik seçimi, doğruluk oranı, WEKA.

ABSTRACT

In this study, the topic of feature selection, which is featured in many publications on software fault prediction in the literature, has been investigated. Feature selection is generally used to increase the accuracy of the classifier by reducing irrelevant and unnecessary features in datasets. In the study, various feature extraction methods were tested on NASA datasets and an experimental dataset, and the operations were performed using the two most suitable methods, Cfs Subset Eval algorithm and Principal Component feature selection methods. As a result, an attempt was made to determine which algorithms have higher success rates.

When the obtained results were examined, an improvement in accuracy rates was generally observed, while some algorithms showed only a minimal difference. When different feature extraction methods were tested on the JM1, KC1, CM1, and PC1 datasets, the 22 features present in all datasets were reduced to 8 features by selecting the most appropriate methods, namely the Cfs Subset Eval algorithm and Principal Component feature selection methods. Subsequently, the accuracy rates of 46 classification algorithms were calculated on the WEKA platform. The best changes in accuracy rates across all datasets were observed with the Bayes Net, Voted Perceptron, K*, and Random Forest algorithms.

It was observed that the loc, n, v, and defect features of the software metrics should definitely be included in all feature selection methods applied on the NASA datasets and experimental datasets. It is clear that the loc (lines of code), n (total number of distinct operators and distinct operands), v (program volume), and defect (whether there is a fault or not) features are quite important in the calculation of software metrics that constitute each dataset.

Keywords: Machine learning algorithms, software defect prediction, feature seletion, accuracy score, WEKA.

1. GİRİŞ

Öznitelik seçimi, genellikle ilgisiz ve gereksiz öznitelikleri azaltarak sınıflandırıcının doğruluğu arttırmak amaçlı kullanılır ve veri setindeki boyutların azaltılması için verilerin ön işlemden geçirilmesi aşamalarındandır. Veriler üzerinde herhangi bir dönüşüm yapmadan, mevcut özelliklerin bir alt kümesini seçmektedir. Bu çalışmada yazılım hata tahmini konusunda literatürde yayınlanmış birçok yayında yer alan öznitelik seçimi konusu araştırılmıştır.

Özniteliklerin önemini bulmak için öznitelik seçim algoritmaları kullanılır. Öznitelik seçimi, hangi makine öğrenmesi algoritmalarında hangi özniteliklerin daha önemli olduğu ve hangi öznitelik seçimlerinde doğruluk oranlarının daha başarılı sonuçlar verdiğini ortaya koymaktadır. Özellik seçiminde kullanılan tek bir yöntem yoktur. Kullanılacak olan yöntem veri setinin durumuna göre değişkenlik gösterebilir. Farklı öznitelik seçim algoritmaları literatürde mevcuttur. Bunlar, Cfs Subset Eval, Korelasyon Niteliği, OneR, Kazanç Oranı, Bilgi Kazancı, relief-F ve Temel Bileşen gibi algoritmalardır.

Bu çalışmada NASA veri setleri ve deneysel veri seti üzerinde öznitelik seçimi algoritmaları üzerinde çeşitli işlemler gerçekleştirilmiş ve veri setlerinde yer alan hangi özniteliklerin daha önemli, hangi özniteliklerin daha önemsiz olduğunun tespit edilmesi sağlanmıştır. Bunun sonucu olarak hangi algoritmaların başarı oranlarının daha yüksek olduğu tespit edilmeye çalışılmıştır. Elde edilen sonuçlar incelendiğinde, genellikle doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark görülmektedir.

Bu çalışmada farklı özellik çıkarım metotları denenerek, seçilen en uygun iki tanesi olan Cfs Subset Eval algoritması ve Principal Component özellik seçim metotları ile veri setleri üzerinde işlemler gerçekleştirilmiştir

Öznitelik seçimlerinde loc, v(g), lOcomment, lOblank öznitelikleri hem Cfs Subset Eval metotunda hem de Principal Component metotunda tüm makine öğrenmesi algoritmalarının doğruluk oranlarını ölçmek için kesinlikle olması gereken en önemli öznitelik olarak belirlenmiştir.

Deneysel veri seti için, Cfs Subset Eval ve Greedy Stepwise algoritmaları birlikte seçildiğinde sadece loc özniteliğinin seçilmesi durumu ortaya çıkmıştır. Ancak loc özniteliğinin tek başına seçilmesi veri seti için makine öğrenmesi algoritmalarının doğruluk oranı hesabında mantıklı bir yaklaşım olmaz.

Bu yüzden Cfs Subset Eval ve Greedy Stepwise öznitelik seçim algoritmaları deneysel veri setinde kullanılmamıştır.

Deneysel veri setleri üzerinde yapılan farklı özellik çıkarım metotlarında ise, sadece loc özelliği seçilemeyeceği için Cfs Subset Eval algoritmasında öznitelik seçimi uygulanamayacağı görülmüştür. Bu yüzden yalnızca Temel Bileşen özniteliğe seçim metotları seçilmiştir. Veri setlerindeki 14 öznitelik bu metotun Ranker algoritması ile birlikte uygulanması sonucunda 5 özniteliğe düşürülmüş, daha sonra WEKA platformunda 46 adet sınıflandırma algoritmasının doğruluk oranları hesaplanmıştır. Sonuçlara bakıldığında deneysel veri setlerinde doğruluk oranlarındaki en iyi değişimin Multilayer Perceptron, OneR ve Random Tree algoritmalarında olduğu görülmüştür.

NASA veri setleri ve deneysel veri setleri üzerinde uygulanan tüm öznitelik seçim metotlarında yazılım metriklerine ait loc, n, v ve defect özniteliklerinin kesinlikle olması gerektiği görülmüştür. Her veri setini oluşturan yazılım metriklerinin hesaplamasında loc (kod satır sayısı), n(tekil operatör ve tekil operand sayısı toplamı), v(program hacmi) ve defect (hata olup olmadığı) özniteliklerinin oldukça önemli olduğu açıkça belli olmuştur.

2. MALZEME VE YÖNTEM

2.1 Veri Setleri

2.1.1 NASA Veri Setleri

Bu çalışmada yazılım hata tahmini konusunda en fazla kullanılan PROMISE veri tabanında yer alan ve NASA'ya ait yazılım projelerindeki bilgileri içeren JM1, PC1, KC1, CM1 adlı dört veri seti incelenmiştir. Literatürdeki diğer modellerle kıyaslama imkânı oluşturması, veri setlerinin tüm araştırmacılara ücretsiz ve açık bir şekilde sunulması, oldukça detaylı şekilde bilgilerin oluşturulmuş olmasından dolayı projelerdeki hata bilgileri ve metrikleri kullanılmıştır. Veri setleri, metrik ölçüm değerlerini ve değişkenleri içerir. Veri setindeki kayıtların her birinin bir sınıf etiketi vardır, bu da bağlı olduğu yazılım modülünde bildirilen veya rapor edilmeyen bir hata olduğu anlamına gelir. Tablo 2.1'de, veri setlerinin adları, sahip oldukları özelliklerin sayısı, kayıt sayısı, kullanılan programlama dili ve içeriği verilmektedir. Veri setleri özellikleri de aşağıda sunulmuştur.

Tablo 2.1: Veri Setlerinin Özellikleri

Adı	Özellik Sayısı	Metot Sayısı	Programlama Dili	İçerik
JM1	22	10.885	C	Yer Sistemi Bilgileri
KC1	22	2.109	C++	Lokasyon Depo Yönetimi Bilgileri
CM1	22	498	C	Uzay Araç Bilgileri
PC1	22	1.109	C	Uçuşlardaki Dünya Yörünge Bilgileri

JM1: C dili ile geliştirilmiş, gerçek zamanlı bir projedir. 315.000 kod satırı ve 10.885 fonksiyon içermektedir. 8 yıllık hata bilgisi içermektedir ve modüllerde hata raporlandıkça düzeltmeler yapılarak güncellenmektedir. NASA'da yer alan en kapsamlı veri setlerindendir. Bu veri setindeki modüllerin %19'u test ya da sahada hataya neden olmuştur.

KC1: C++ dili ile geliştirilmiştir ve 750.000 kod satırı 2.109 modülden oluşmaktadır. 5 yıllık hata verisi içermekte olan veri setindeki modüllerin %15'i test ya da sahada hataya neden olmuştur. Kod satırı olarak JM1'den geniş olsa da modül ve içerdiği hata miktarı ile daha küçük bir veri setidir.

CM1: C dili ile geliştirilmiş, 20.000 kod satırı ve 498 modüle sahip, 2 yıllık hata bilgisi içeren ve modüllerinin %10'u hatalı olan bir projedir.

PC1: C dili ile geliştirilmiş, 40.000 satır ve 1109 modüle sahip, 3 yıllık hata bilgisi içeren ve bu modüllerin %7'si test ya da sahada hataya neden olan bir veri setidir (Çatal,2008)

Tablo 2.2 veri setlerinde kullanılan McCabe ve Halstead ölçümlerinin özelliklerini açıklamaktadır.

Tablo 2.2: Metrik Özellikleri

Özellik	Açıklama
loc	McCabe kod satır sayısı"
v(g)	McCabe "çevrimsel karmaşıklık"
ev(g)	McCabe "temel karmaşıklık"
iv(g)	McCabe "tasarım karmaşıklığı"
n	Halstead toplam operatör +operand
v	Halstead "yoğunluk"(volume)
l	Halstead "program uzunluğu"(level)
d	Halstead "zorluk"(difficulty)
i	Halstead "zeka"(intelligence)
e	Halstead "çaba"(effort)
b	Halstead "hata"(bug)
t	Halstead "zaman tahmincisi"(time)
lOCode	Halstead "satır sayısı"
lOComment	Halstead "yorum satır sayısı"
lOBlank	Halstead "boş satır sayısı"
uniq_Op	Benzersiz operatörler
uniq_Opnd	Benzersiz operandlar
total_Op	Toplam operatörler
total_Opnd	Toplam operandlar
branchCount	Akış Grafiğindeki Şube Sayısı
defects	{false,true} Modülde rapor edilmiş bir veya birden fazla hata sayısı

2.1.2. Deneysel Veri Seti

Çalışmanın bu bölümünde, Bilgisayar Mühendisliği bölüm öğrencilerine ait C++ dilinde geliştirilen 102 tane yazılım projelerindeki program kodları kullanılarak elde edilmiş yazılım metrikleri deneysel bir veri seti haline getirilmiştir. Veri setindeki 102 proje kodu içinde 72 öğrencinin kodunda hata bulunmakta, 30 öğrencinin kodunda ise hata bulunmamaktadır. Bu veri setini oluşturan yazılım metriklerinin seçiminde NASA veri setlerinde ortak olan ve literatürde de sıklıkla kullanılan metrik ölçüm değerleri kullanılmıştır. Tablo 2.3'de deneysel veri setinin sahip oldukları özellikler, kullanılan programlama dili ve içeriği verilmektedir. Deneysel veri setini oluşturmak için NASA veri setlerinde de ortak olarak yer alan 14 metrik seçilmiştir. Tablo 2.4'te deneysel veri setine ait metriklerin özellikleri yer almaktadır.

Tablo 2.3: Deneysel Veri Seti Özellikleri

Adı	Özellik Sayısı	Satır Sayısı	Programlama Dili	İçerik
Öğrenci Veri Seti	14	9.142	C++	Öğrenci Yazılım Proje Kodları

Tablo 2.4: Deneysel Veri Seti Metrik Özellikleri

Özellik	Açıklama
loc	McCabe kod satır sayısı
n	Halstead Kelime Sayısı
N	Halstead Program Uzunluğu
v	Halstead "yoğunluk" (volume)
d	Halstead "zorluk" (difficulty)
e	Halstead "çaba"(effort)
b	Halstead "hata" (bug)
t	Halstead “zaman tahmincisi” (time)
lOComment	Halstead “yorum satır sayısı”
uniq_Op	Benzersiz operatörler
uniq_Opnd	Benzersiz operandlar
total_Op	Toplam operatörler
total_Opnd	Toplam operandlar
defects	{1,0} Modülde rapor edilmiş bir veya birden fazla hata sayısı

Veri setini oluşturma aşamasında öncelikle ilk adım olarak 102 adet yazılım projesi için her yazılım koduna ait operatör ve operand sayıları hesaplamıştır. Operatörler bir yazılım kodunda yer alan aritmetik ve mantıksal işlemleri gerçekleştiren (for, if, else if, while vb) gibi anahtar kelimeler ve semboller olarak adlandırılır. Operandlar ise, aritmetik ve mantıksal işlemlerde kullanılan diğer değişkenler ve sabit sayılara verilen addır. Yazılım proje kodlarında yer alan bu operatörler ve operandlar sayısal olarak toplanarak diğer yazılım metriklerinin hesaplanmasında temel girdi olarak rol oynar. Veri setini oluşturmak için 102 yazılım kodunda yer alan tüm operatör ve operandlar teker teker sayılarak kaç adet oldukları, isim etiketlendirmesi ile beraber bir dosyaya kaydedilmiştir. Öncelikle 102 tane yazılım proje kodlarında kullanılan benzersiz (tekil) operatör sayısı ve toplam operatör sayıları hesaplanmış ve oluşturulmuş her bir projeyi temsilen proje kod numarası ile birlikte tabloya eklenmiştir. Daha sonra ise benzersiz (tekil) operand ve toplam operand sayısı hesaplanmış ve benzer şekilde proje kod numaraları ile birlikte tabloya eklenmiştir. Hesaplanan benzersiz operatör- toplam operatör ve benzersiz operand- toplam operand sayısal bilgileri deneysel veri seti için kullanılacak olan temel metriklerdir

2.1 Öznitelik Seçimi

Öznitelik seçimi bir veri seti içinden, modelin başarısını etkileyen gerekli verilerin seçilip gereksiz verilerin çıkarılması işlemidir. Bu şekilde modelin başarısı artırılmaktadır. Öznitelik seçimi veriler üzerinde herhangi bir dönüşüm yapmadan mevcut özniteliklerin bir alt kümesini oluşturmaktadır (Güven Aydın, 2021). Öznitelik seçimi, sınıflandırma sistemlerinde verimli ve yaygın bir şekilde kullanılmaktadır. Öznitelik seçimi ile yapılan sınıflandırmada, işlem sayısı azalır, gürültülü ve alakasız öznitelikler veri setinden çıkarılarak sınıflandırma başarısı arttırılır. Eğitim zamanı kısalır, daha az ölçüm yapılır ve daha az bellek tüketilir. Bu sayede, anlamlı ve daha kolay sınıflandırma sağlanmış olur (Abe vd., 1998) , (Huang ve Chow, 2005).

2.2 Öznitelik Seçim Algoritmaları

Bu çalışmada WEKA programında bulunan CfsSubsetEval (Corelation-based Feature Subset Selection Evaluation – Korelasyon Tabanlı Özellik Seçim Değerlendirici) yöntemi, en etkili özniteliklerin ortaya çıkarılması amacıyla kullanılmıştır. CfsSubsetEval, öznitelik alt kümelerini

korelasyon değerine göre sıralayan sezgisel basit bir filtre algoritmasıdır. En iyi öznitelik alt kümesini korelasyon yardımı ile bulmaktadır. Bu algoritma sınıfla yüksek düzeyde ilişkili olan ve birbirleriyle ilişkisiz öznitelikler içeren alt kümeleri değerlendirmektedir. Alakasız olan öznitelikler sınıfla düşük korelasyona sahip olacağından göz ardı edilmektedir. CfsSubsetEval bir arama yöntemi değildir, bunun yerine arama algoritmalarına öznitelik alt kümesinin etkinliğini değerlendirmek için bir metrik önermektedir. Algoritmanın temelinde çıktı sınıfıyla yüksek oranda ilişkili ancak birbiriyle ilişkisiz özelliklere sahip iyi bir öznitelik alt kümesi oluşturma yatmaktadır (Hall, 1999). CfsSubsetEval, herhangi bir açgözlü veya meta-sezgisel arama yaklaşımıyla kullanılabilir.

2.2.1. CFS Subset Eval Öznitelik Seçim Algoritması

Cfs Subset öznitelik seçim algoritması, en iyi öznitelikleri veri setindeki değişkenleri korelasyon yardımı ile bulur. Bu algoritma aralarında düşük korelasyonlu, sınıf etiketleri arasında ise yüksek korelasyonlu öznitelikleri seçer. Bu algoritmada, yüksek korelasyonlu özellikler veri setinden çıkarılır çünkü ilişkisiz özellikler daha iyi sınıflandırma başarısı ortaya çıkarır. Algoritma, özniteliklerin öngörü yeteneği ve fazlalık derecesi temelinde önemini ölçer. Daha az karşılıklı korelasyona sahip olan ancak hedef sınıfla yüksek oranda ilişkili olan alt gruplar tercih edilir.

Cfs Subset bir arama algoritması kullanmaktadır. Algoritmada nitelikler arasından, sınıf etiketi ile en iyi ilişkiye sahip olanların belirlenmesi sağlanmaktadır. Bu durumda belirlenen özellik grubu sınıf etiketi ile yüksek bir bağlantı içermekte fakat diğer niteliklerin daha önemsiz bir duruma sahip olduğu tespit edilmektedir. Dolayısıyla sınıf etiketi ile yüksek bağlantıya sahip olan nitelik grubunun kullanımının başarımı artırması beklenmektedir. (Gümüşçü, İ. B. Aydilek ve R. Taşaltın, 2016). Bu çalışmada veri setleri üzerinde Cfs Subset Eval algoritması ile birlikte Greedy Stepwise arama algoritması birlikte kullanılmıştır.

2.2.2. Temel Bileşen Öznitelik Seçim Algoritması

Verilerin temel bileşen analizi ve dönüşümünü gerçekleştirir. Ranker arama algoritmasıyla birlikte kullanılır. Veri setinin, kovaryans matrisinin veya tekil değer çıkarımının yöntemi ile elde edilen basitleştirilmiş halidir. Algoritma, sadece veri setlerini sadeleştirmez, verilerin birbiri ile olan ilişkisini de ortaya koyar. Böylece verilerin sonuca etkilerinin ağırlıklarının hesaplanması için de kullanılır. Amacı, orijinal öznitelikler alanını özniteliklerin ilişkisiz olduğu yeni bir alana dönüştürerek çok sayıda ilişkili öznitelik içeren veri setinin boyutluluğunu azaltmaktır. Algoritma daha sonra orijinal veri seti ile yenisi arasındaki değişimi sıralar.

3. BULGULAR

3.1 NASA Veri Setleri ve Deneysel Veri Seti ile Öznitelik Seçim Algoritması

Çalışmasının bu bölümünde NASA veri setleri ve deneysel veri seti üzerinde öznitelik seçimi algoritmaların, metriklerden oluşan veri setlerinde yer alan hangi özellikler bir arada kullanıldığında veya hangi metriklerin önemli, hangi metriklerin o algoritma için önemsiz olduğunun bunun sonucu olarak da hangi algoritmaların başarı oranlarının daha yüksek olduğu öznitelik seçim algoritmalarıyla belirlenmiştir.

JM1 veri seti için, Cfs Subset Eval ve Greedy Stepwise algoritmaları birlikte seçildiğinde loc, $v(g), ev(g), iv(g), i,\ lOcomment, lOblank, lOcodeandcomment$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları gerçeklendiğinde bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.5.'te verilmiştir. Tablo incelendiğinde, doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark görülmektedir. JM1 veri seti için, Principal

Components ve Ranker algoritmaları birlikte seçildiğinde $loc, v(g), ev(g), iv(g), n, v, l, d$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları gerçeklendiğinde bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.5.'te verilmiştir. Tablo incelendiğinde, bazı algoritmalarda doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmaların doğruluk oranlarında çok az bir fark görülmektedir. Tabloda sırasıyla *: Cfs Subset Eval+Best First (Korelasyon Tabanlı Özellik Seçici + Önce En İyi (Doğruluk Oranı %) (Özellik Seçimi Olmadan), **: Cfs SubsetEval+Greedy Stepwise (Korelasyon Tabanlı Özellik Seçici + Açgözlü) (Doğruluk Oranı %) ve ***: Principal Componenets+Ranker (Temel Bileşen+ Sıralayıcı) (Doğruluk Oranı %) ifade etmektedir.

KC1 veri seti için, Cfs Subset Eval ve Greedy Stepwise algoritmaları birlikte seçildiğinde $v, d, i, lOcode, lOcomment, lOblank, uniqopnd, branchcount$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları gerçeklendiğinde bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.6'da verilmiştir. Doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark görülmektedir. KC1 veri seti için, Principal Components ve Ranker algoritmaları birlikte seçildiğinde $loc, v(g), ev(g), iv(g), n, v, l, d$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları gerçeklendiğinde bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.6'da verilmiştir. Tablo incelendiğinde, doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark görülmektedir.

CM1 veri seti için, Cfs Subset Eval ve Greedy Stepwise algoritmaları birlikte seçildiğinde $loc, i(v), i, lOcomment, lOblank, uniqop, uniqopnd,$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.7'de verilmiştir. Tablo incelendiğinde, doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark görülmektedir. CM1 veri seti için, Principal Components ve Ranker algoritmaları birlikte seçildiğinde $loc, v(g), ev(g), iv(g), n, v, i$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları gerçeklendiğinde bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.7'de verilmiştir. Tablo incelendiğinde, doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark görülmektedir.

PC1 veri seti için, Cfs Subset Eval ve Greedy Stepwise algoritmaları birlikte seçildiğinde v(g), $i, lOcomment, lOcodeandcomment, lOblank, uniq\_opnd$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları gerçeklendiğinde bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.8'de verilmiştir. Tablo incelendiğinde, doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark görülmektedir. PC1 veri seti için, Principal Components ve Ranker algoritmaları birlikte seçildiğinde $loc, v(g), ev(g), iv(g), n, v, i$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları gerçeklendiğinde bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.8 'da verilmiştir. Tablo incelendiğinde, doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmalarda çok az bir fark görülmektedir.

JM1, KC1, CM1 ve PC1 veri setleri üzerinde farklı özellik çıkarım metotları test edildiğinde, tüm veri setlerinde yer alan 22 özellik, en uygun metotların Cfs Subset Eval algoritması ve Principal Component özellik seçim metotlarının seçilmesiyle birlikte 8 özellliğe düşürülmüştür. Daha sonra WEKA platformunda 46 adet sınıflandırma algoritmasının doğruluk oranları hesaplanmıştır. Tüm veri setlerinde doğruluk oranlarında en iyi değişim Bayes Net, Voted Perceptron, K* ve Random Forest algoritmalarında görülmüştür.

Öznitelik seçimlerinde $loc, v(g), lOcomment, lOblank$ öznitelikleri hem Cfs Subset Eval metotunda hem de Principal Component metotunda tüm makine öğrenmesi algoritmalarının doğruluk oranlarını ölçmek için kesinlikle olması gereken en önemli özellik olarak belirlenmiştir.

Deneysel veri seti için, Cfs Subset Eval ve Greedy Stepwise algoritmaları birlikte seçildiğinde sadece *loc* özelliğinin seçilmesi durumu ortaya çıkmıştır. Ancak loc özelliğinin tek başına seçilmesi veri seti için makine öğrenmesi algoritmalarının doğruluk oranı hesabında mantıklı bir yaklaşım olmadığı

değerlendirilmiştir. Bu yüzden Cfs Subset Eval ve Greedy Stepwise öznitelik seçim algoritmaları deneysel veri setinde kullanılmamıştır.

Deneysel veri setinde, Principal Components ve Ranker algoritmaları birlikte seçildiğinde $loc, v(g), n, N, V$ ve defect öznitelikleri seçilip makine öğrenmesi algoritmaları gerçeklendiğinde bazı algoritmalarda doğruluk oranlarında elde edilen değişim Tablo 3.9'da verilmiştir. Tablo incelendiğinde, bazı algoritmalarda doğruluk oranlarında bir iyileşme oranı görülürken, bazı algoritmaların doğruluk oranlarında çok az bir fark görülmektedir. Tabloda sırasıyla *: Cfs Subset Eval+Best First (Korelasyon Tabanlı Özellik Seçici + Önce En İyi (Doğruluk Oranı %) (Özellik Seçimi Olmadan), ve **: Principal Componenets+ Ranker (Temel Bileşen+ Sıralayıcı) (Doğruluk Oranı %) ifade etmektedir.

Tablo 3.5: JM1 Veri Seti Öznitelik Seçimi ile Doğruluk Oranı Değişimi

Algoritma	*	**	***
Bayes Net	68,05	75,71	72,00
Naive Bayes	80,42	80,41	80,31
Naive Bayes Multinomial		71,74	78,62
Naive Bayes Multinominal Text	80,65	80,65	80,65
Naive Bayes Multinomial Updateable		71,74	78,62
Naive Bayes Updatable	80,42	80,41	80,31
Logistic	81,35	80,95	81,08
Multilayer Perceptron	80,95	81,10	81,12
Stochastic Gradient Descent	80,77		80,77
Stochastic Gradient Descent Text	80,65	80,65	80,65
Simple Logistic	81,12	80,86	81,13
Sequential Minimal Optimization	80,72	80,65	80,68
Voted Perceptron	52,21	79,55	80,87
K-Nearest Neighbours Classifier	76,97	76,52	76,86
K*	78,56	80,96	80,91
Locally Weighted Learning	80,89	80,65	80,65
Iterative Classifier Optimizer	80,89	80,95	81,01
Adaboost1	80,79	80,65	80,82
Attibute Selected Classifier	80,86	81,0	81,14
Bagging	81,19	81,11	81,22
Classification via Regression	81,24	80,80	81,14
CVParameter Selection	80,65	80,65	80,65
Filtered Classsifier	81,12	81,13	81,40
Logi Boost	80,89	80,65	81,01
Multi Class Classifier	81,35	80,95	81,08
Multi Class Classifier Updatable	80,77	80,71	80,77
Multi Scheme	80,65	80,65	80,65
Random Comittee	81,03	79,54	79,95
Randomizable Filtered Classifer	75,26	76,15	75,50
Random Sub Space	81,49	81,31	81,43
Stacking	80,65	80,65	80,65
Vote	80,65	80,65	80,65
Weighted Instances Handler Wrapper	80,65	80,65	80,65
Input Mapped Classifier	80,65	80,65	80,65
Desicion Table	80,90	80,80	80,97
Repeated Incremental Pruning	81,04	81,15	81,16
OneR	79,39	79,88	79,71

Algoritma	*	**	***
Part	80,74	80,98	81,25
ZeroR	80,65	80,65	80,65
Desicion Stump	80,65	80,65	80,65
Hoeffding Tree	80,71	80,46	80,65
J48	79,50	81,00	80,93
LMT	81,24	80,91	81,20
Random Forest	81,75	81,20	81,19
Random Tree	75,47	75,66	75,06
Rep Tree	80,67	81,17	80,76

Tablo 3.6: KC1 Veri Seti Öznitelik Seçimi ile Doğruluk Oranı Değişimi

Algoritma	*	**	***
Bayes Net	69,89	75,77	76,57
Naive Bayes	82,36	82,40	82,88
Naive Bayes Multinomial	85,34	83,40	82,83
Naive Bayes Multinominal Text	84,54	84,54	84,54
Naive Bayes Multinomial Updateable	85,34	83,40	82,83
Naive Bayes Updatable	82,36	82,40	82,88
Logistic	85,68	85,20	84,63
Multilayer Perceptron	85,91	85,72	85,01
Stochastic Gradient Descent	85,20	84,44	84,49
Stochastic Gradient Descent Text	84,54	84,54	84,54
Simple Logistic	85,72	85,15	85,06
Sequential Minimal Optimization	84,77	84,49	84,54
Voted Perceptron	83,73	84,40	84,54
K-Nearest Neighbours Classifier	84,40	84,30	83,30
K*	83,97	86,29	86,53
Locally Weighted Learning	71,72	84,73	84,54
Iterative Classifier Optimizer	85,25	85,30	85,82
Adaboost1	84,96	84,92	84,49
Attibute Selected Classifier	84,30	84,16	84,11
Bagging	86,01	85,01	84,87
Classification via Regression	85,58	85,44	85,30
CVParameter Selection	84,54	84,54	84,54
Filtered Classsifier	84,87	84,54	85,15
Logi Boost	85,39	85,30	85,72
Multi Class Classifier	85,68	85,20	84,63
Multi Class Classifier Updatable	85,20	84,44	84,49
Multi Scheme	84,54	84,54	84,54
Random Comittee	85,39	84,82	84,68
Randomizable Filtered Classifer	81,31	83,92	83,07
Random Sub Space	85,49	85,20	85,11
Stacking	84,54	84,54	84,54
Vote	84,54	84,54	84,54
Weighted Instances Handler Wrapper	84,54	84,54	84,54
Input Mapped Classifier	84,54	84,54	84,54
Desicion Table	84,87	84,92	85,25
Repeated Incremental Pruning	84,54	85,06	84,58
OneR	83,68	83,68	82,78
Part	84,82	84,87	84,63

Algoritma	*	**	***
ZeroR	84,54	84,54	84,54
Desicion Stump	84,54	84,54	84,54
Hoeffding Tree	84,54	84,54	84,54
J48	84,54	84,68	84,54
LMT	85,72	85,15	85,15
Random Forest	86,67	85,44	85,58
Random Tree	82,69	82,78	83,45
Rep Tree	85,11	84,54	84,54

Tablo 3.7: CM1 Veri Seti Öznitelik Seçimi ile Doğruluk Oranı Değişimi

Algoritma	*	**	***
Bayes Net	64,65	75,50	74,89
Naive Bayes	85,34	86,54	86,74
Naive Bayes Multinomial	70,68	85,34	86,14
Naive Bayes Multinominal Text	90,16	90,16	90,16
Naive Bayes Multinomial Updateable	70,68	85,34	86,14
Naive Bayes Updatable	85,34	86,54	86,74
Logistic	88,35	89,95	88,75
Multilayer Perceptron	87,55	89,15	89,55
Stochastic Gradient Descent	89,55	90,16	90,16
Stochastic Gradient Descent Text	90,16	90,16	90,16
Simple Logistic	89,15	89,75	89,75
Sequential Minimal Optimization	89,55	90,16	90,16
Voted Perceptron	90,16	89,35	90,16
K-Nearest Neighbours Classifier	84,73	83,13	81,52
K*	87,14	85,14	86,54
Locally Weighted Learning	89,75	89,55	90,16
Iterative Classifier Optimizer	89,15	88,75	90,16
Adaboost1	90,16	90,16	90,16
Attibute Selected Classifier	89,35	89,35	90,16
Bagging	89,75	89,35	89,35
Classification via Regression	89,35	89,75	88,95
CVParameter Selection	90,16	90,16	90,16
Filtered Classsifier	90,16	90,16	90,16
Logi Boost	88,95	88,55	89,55
Multi Class Classifier	88,35	89,95	88,75
Multi Class Classifier Updatable	89,55	90,16	90,16
Multi Scheme	90,16	90,16	90,16
Random Comittee	87,75	88,15	86,94
Randomizable Filtered Classifer	85,54	84,33	84,13
Random Sub Space	90,16	90,16	90,16
Stacking	90,16	90,16	90,16
Vote	90,16	90,16	90,16
Weighted Instances Handler Wrapper	90,16	90,16	90,16
Input Mapped Classifier	90,16	90,16	90,16
Desicion Table	89,15	89,15	90,16
Repeated Incremental Pruning	89,35	87,95	89,95
OneR	88,35	88,35	88,35
Part	88,75	88,75	89,15
ZeroR	90,16	90,16	90,16

Algoritma	*	**	***
Desicion Stump	90,16	90,16	90,16
Hoeffding Tree	90,16	90,16	90,16
J48	87,95	89,35	89,55
LMT	89,15	88,55	89,15
Random Forest	88,75	88,55	88,35
Random Tree	84,33	84,73	84,13
Rep Tree	89,15	89,55	90,16

Tablo 3.8: PC1 Veri Seti Öznitelik Seçimi ile Doğruluk Oranı Değişimi

Algoritma	*	**	***
Bayes Net	74,39	87,19	85,12
Naive Bayes	89,17	89,99	89,72
Naive Bayes Multinomial	90,53	88,99	88,54
Naive Bayes Multinominal Text	93,05	93,05	93,05
Naive Bayes Multinomial Updateable	90,53	88,99	88,54
Naive Bayes Updatable	89,17	89,99	89,72
Logistic	92,42	93,14	92,87
Multilayer Perceptron	93,59	93,05	92,96
Stochastic Gradient Descent	93,05	93,05	93,05
Stochastic Gradient Descent Text	93,05	93,05	93,05
Simple Logistic	92,60	92,87	92,87
Sequential Minimal Optimization	92,96	93,05	93,05
Voted Perceptron	92,60	93,05	93,05
K-Nearest Neighbours Classifier	92,06	91,07	91,43
K*	91,79	92,42	92,33
Locally Weighted Learning	93,23	93,05	93,05
Iterative Classifier Optimizer	93,05	93,05	92,96
Adaboost1	93,05	93,05	93,05
Attibute Selected Classifier	93,41	92,96	92,96
Bagging	94,13	92,96	92,87
Classification via Regression	93,14	93,14	92,96
CVParameter Selection	93,05	93,05	93,05
Filtered Classsifier	93,50	93,05	93,05
Logi Boost	93,14	92,96	92,78
Multi Class Classifier	92,42	93,14	92,87
Multi Class Classifier Updatable	93,05	93,05	93,05
Multi Scheme	93,05	93,05	93,05
Random Comittee	93,59	92,51	92,33
Randomizable Filtered Classifer	89,45	92,15	90,44
Random Sub Space	93,86	93,14	93,14
Stacking	93,05	93,05	93,05
Vote	93,05	93,05	93,05
Weighted Instances Handler Wrapper	93,05	93,05	93,05
Input Mapped Classifier	93,05	93,05	93,05
Desicion Table	92,87	93,05	93,05
Repeated Incremental Pruning	93,32	92,51	93,14
OneR	92,87	92,60	92,60
Part	93,68	93,14	93,14
ZeroR	93,05	93,05	93,05
Desicion Stump	93,05	93,05	93,05
Hoeffding Tree	93,05	93,05	93,05
J48	93,32	92,96	92,87
LMT	92,42	92,87	92,87
Random Forest	93,68	93,32	93,23
Random Tree	91,07	90,89	91,52
Rep Tree	93,59	92,78	93,05

Tablo 3.9: Deneysel Veri Seti Öznitelik Seçimi ile Doğruluk Oranı Değişimi

Algoritma	*	**
BAYES		
Bayes Net	60,78	60,78
Naive Bayes	60,78	60,78
Naive Bayes Multinomial	59,80	57,84
Naive Bayes Multinominal Text	60,78	60,78
Naive Bayes Multinomial Updateable	59,80	57,84
Naive Bayes Updatable	60,78	60,78
FUNCTIONS		
Logistic	55,88	58,82
Multilayer Perceptron	52,94	63,72
Stochastic Gradient Descent	61,76	60,78
Stochastic Gradient Descent Text	60,78	60,78
Simple Logistic	57,84	58,82
Sequential Minimal Optimization	62,75	59,80
Voted Perceptron	56,86	60,78
LAZY		
K-Nearest Neighbours Classifier	51,96	44,11
K*	49,01	42,15
Locally Weighted Learning	62,75	60,78
META		
Iterative Classifier Optimizer	60,78	59,80
Adaboost1	60,78	55,88
Attibute Selected Classifier	59,80	60,78
Bagging	49,01	48,03
Classification via Regression	59,80	62,74
CVParameter Selection	60,78	60,78
Filtered Classsifier	60,78	60,78
Logi Boost	50,98	55,88
Multi Class Classifier	55,88	58,82
Multi Class Classifier Updatable	61,76	60,78
Multi Scheme	60,78	60,78
Random Comittee	45,09	57,84
Randomizable Filtered Classifer	45,09	40,19
Random Sub Space	60,78	60,78
Stacking	60,78	60,78
Vote	60,78	60,78
Weighted Instances Handler Wrapper	60,78	60,78
MISC		
Input Mapped Classifier	60,78	60,78
Desicion Table	59,80	60,78
Repeated Incremental Pruning	55,88	57,84
OneR	46,07	53,92
Part	61,76	60,78
ZeroR	60,78	60,78
TREE		
Desicion Stump	60,78	60,78
Hoeffding Tree	57,84	60,78
J48	61,76	60,78
LMT	57,84	59,80
Random Forest	48,03	50,62
Random Tree	50,98	58,82
Rep Tree	58,82	59,80

4. SONUÇ VE ÖNERİLER

Bu çalışmada literatürde de oldukça fazla kullanılan öznitelik seçim metotları NASA veri setleri üzerinde ve deneysel veri setleri üzerinde kullanılmıştır. NASA'ya ait JM1, KC1, CM1 ve PC1 veri setleri üzerinde farklı özellik çıkarım metotları test edilmiş, en iyi sonuçların Cfs Subset Eval

algoritması ve Temel Bileşen özellik seçim metotlarının seçilmesiyle olduğu görülmüştür. Veri setlerindeki 22 özellik bu metotların uygulanması sonucunda 8 özelliğe düşürülmüş, daha sonra WEKA platformunda 46 adet sınıflandırma algoritmasının doğruluk oranları hesaplanmıştır. Sonuçlara bakıldığında tüm veri setlerinde doğruluk oranlarındaki en iyi değişimin Bayes Net, Voted Perceptron, K* ve Random Forest algoritmalarında olduğu görülmüştür. Öznitelik seçimlerinde $loc, v(g), IOcomment, IOblank$ öznitelikleri hem CfsSubsetEval metotunda hem de Temel Bileşen metodunda tüm makine öğrenmesi algoritmalarının doğruluk oranlarını ölçmek için kesinlikle olması gereken en önemli özellik olarak belirlenmiştir.

Deneysel veri setleri üzerinde yapılan farklı özellik çıkarım metotlarında ise, sadece *loc* özelliği seçilemeyeceği için Cfs Subset Eval algoritmasında öznitelik seçimi uygulanamayacağı görülmüştür. Bu yüzden yalnızca Temel Bileşen özellik seçim metotları seçilmiştir. Veri setlerindeki 14 özellik bu metotun Ranker algoritması ile birlikte uygulanması sonucunda 5 özelliğe düşürülmüş, daha sonra WEKA platformunda 46 adet sınıflandırma algoritmasının doğruluk oranları hesaplanmıştır. Sonuçlara bakıldığında deneysel veri setlerinde doğruluk oranlarındaki en iyi değişimin Multilayer Perceptron, OneR ve Random Tree algoritmalarında olduğu görülmüştür.

KAYNAKÇA

Çatal Ç., (2008). Yazılım Kusur Kestirimi Probleminde Yapay Bağışıklık Sistemlerinin Uygulanması, Doktora, Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü.

Güven Aydın, Z. B. (2021). Makine Öğrenmesi Yöntemleri İle Yazılım Hata Tahmini, Doktora Tezi, İstanbul Üniversitesi-Cerrahpaşa, Lisansüstü Eğitim Fakültesi

Abe, S., Thawonmas, R. and Kobayashi, Y., (1998). Feature selection by analyzing class regions approximated by ellipsoids, IEEE Trans. On Systems, Man, and Cybernetics-Part C: Applications and Reviews, 28(2), 282 – 287.

Huang, D., Chow, T. W. S., (2005). Efficiently searching the important input variables using Bayesian discriminant. IEEE Trans. on Circuits and Systems-I: Regular Papers, 52(4), 785

Hall, Mark A., (1999). Correlation-based Feature Selection for Machine Learning, Doktora Tezi, University of Waikato, Department of Computer Science.

Gümüşçü, İ. B. Aydilek ve R. Taşaltın, "Mikro-dizilim Veri Sınıflandırmasında Öznitelik Seçme Algoritmalarının Karşılaştırılması," *Harran Üniversitesi Mühendislik Dergisi*, 1(1), 1-7, 2016.