Avrupa Bilim ve Teknoloji Dergisi
Sayı 27, S. 985-994, Kasım 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 27, pp. 985-994, November 2021

Research Article

Genetik Algoritma İle Öznitelik Seçimi Yapılarak Yazılım Projelerinin Maliyet Tahmini

Sukran Ebren Kara[1,2,*], Ruya Samli[2]

[1*] Şırnak Üniversitesi, Cizre Meslek Yüksekokulu, Bilgisayar Teknolojileri Bölümü, Şırnak, Türkiye, (ORCID: 0000-0003-3071-6942), sebrenkara@sirnak.edu.tr
[2] İstanbul Üniversitesi-Cerrahpaşa, Bilgisayar Mühendisliği Bölümü İstanbul, Türkiye (ORCID: 0000-0002-8723-1228), ruyasamli@iuc.edu.tr

(İlk Geliş Tarihi 12 Eylül 2021 ve Kabul Tarihi 28 Kasım 2021)

(**DOI:** 10.31590/ejosat.994372)

ATIF/REFERENCE: Ebren Kara, Ş. & Şamlı, R. (2021). Genetik Algoritma İle Öznitelik Seçimi Yapılarak Yazılım Projelerinin Maliyet Tahmini. *Avrupa Bilim ve Teknoloji Dergisi*, (27), 985-994.

Öz

Bir yazılım projesinin tahmini maliyetini, yazılımı geliştirme döngüsünün başlarında yapabilmek proje yöneticisi için çok büyük önem arz etmektedir. Projede ön görülemeyen belirsizlikler, zaman ve maliyet açısından proje yöneticisine sorunlar doğuracaktır. Yazılım maliyetinin doğru tahmin edilmesi bu gibi sorunları en aza indirmektedir. Literatürdeki çalışmalara bakıldığında, yazılım projelerinin maliyetinin çok farklı yöntemlerle tahmin edilmeye çalışıldığı görülmektedir. Bu çalışmanın amacı, bu yöntemler arasında sıklıkla kullanılan bir yöntem olarak ifade edilebilecek olan Genetik Algoritma (GA) kullanılarak veri setlerinde öznitelik seçiminin yazılım projelerinin maliyet tahminine etkisinin araştırılmasıdır.

Bu çalışmada yazılım projelerinin maliyet tahmini için, WEKA (Waikato Environment for Knowledge Analaysis – Bilgi Analizi için Waikato Ortamı) programında bulunan 8 farklı Makine Öğrenmesi algoritması ve Evrimsel Algoritma: Genetik Programlama (GP) (Genetic Programming) varsayılan ayarlar ile iki şekilde çalıştırılmıştır. İlk olarak, PROMISE (Predictor Models in Software Engineering – Yazılım Mühendisliğinde Tahmin Modelleri) veri deposundan temin edilen ham veri setlerine (Albrecht, Finnish, Kemerer, Maxwell ve Miyazaki94) herhangi bir öznitelik seçimi yapılmadan Makine Öğrenmesi algoritmaları uygulanarak yazılım maliyet tahmini gerçekleştirilmiştir. İkinci olarak, öncelikle veri setlerine GA uygulanarak öznitelik seçimi yapılmıştır. Öznitelik seçimi ile ilgili alt küme çıkarıldıktan sonra veri setlerine Makine Öğrenmesi algoritmaları uygulanarak yazılım maliyet tahmini gerçekleştirilmiştir. Algoritmalar 10 kat çapraz doğrulama tekniği ile test edilmiş ve sonuçlar değerlendirilirken, hata oranları, korelasyon katsayısı, MAE (Mean Absolute Error – Ortalama Mutlak Hata) ve RAE (Relative Absolute Error – Bağıl Mutlak Hata) dikkate alınmıştır. Bulgular karşılaştırılıp performans değerleri analiz edildiğinde, GA ile öznitellik seçimi yapılan veri setlerinden elde edilen tahmin sonuçlarının öznitelik seçimi yapılmadan elde edilen tahmin sonuçlarından daha iyi olduğu belirlenmiştir.

Anahtar Kelimeler: Genetik Algoritmalar, Genetik Programlama, Makine Öğrenmesi, Yazılım Maliyet Tahmini, WEKA.

Cost Estimation of Software Projects by Feature Selection with Genetic Algorithm

Abstract

It is very important for the project manager to be able to estimate the cost of a software project early in the software development cycle. The project manager can reduce the uncertainties in the project by accurately estimating the project cost. Otherwise, serious economic problems will arise. Looking at the studies in the literature, it is seen that the cost of software projects has been tried to be estimated using very different methods. The aim of this study is to investigate the effect of feature selection with genetic algorithms on software cost estimation. In this study, 8 different Machine Learning algorithms in the WEKA (Waikato Environment for Knowledge Analysis) environment and Evolutionary Algorithm: Genetic Programming were run in two ways with default settings for the cost estimation of software projects. First, software cost estimation was performed by applying Machine Learning algorithms to the raw data sets (Albrecht, Finnish, Kemerer, Maxwell and Miyazaki94) which were obtained from the PROMISE (Predictor Models in Software Engineering) data store without any feature selection. Secondly, feature selection was made by applying Genetic Algorithm to the

datasets. After the subset of datasets was created by feature selection, Machine Learning algorithms were applied to the data sets and software cost estimation was realized. Algorithms were applied to datasets with 10-fold cross validation technique and results, MAE (Mean Absolute Error) and RAE (Relative Absolute Error) and performance criterion correlation coefficient. When the results were examined and the performance values were compared, it was determined that the estimation results obtained from the data sets with feature selection by Genetic Algorithm were better than the estimation results obtained without feature selection.

Keywords: Genetic Algorithms, Genetic Programming, Machine Learning, Software Cost Estimation, WEKA.

1. Giriş

Yazılım maliyet tahmini, yazılım geliştirme sürecinin en ciddi problemlerinden birisidir. Çünkü yazılım soyut bir üründür ve birçok bilinmeyeni içermektedir. Bundan dolayı yazılım geliştirme süreci hem zordur hem de zaman alıcıdır (Yücalar, 2011). Demirörs (2011) yazılım proje yönetimini, başlangıç, planlama, yürütme, izleme, kontrol ve kapanış aşamalarından oluşan bir süreç olarak tanımlamıştır. Araştırmacı, projenin yürütme aşamasında yöneticilerin projenin durumunu belirlemesi, izlemesi, kontrol ve müdahale edebilmesi için gerçek proje verilerine ihtiyaç duyduklarını ayrıca güvenilir modeller oluşturarak projede gerekli olan kaynakları belirleyebilmesi için geçmiş proje verisine sahip olması gerektiğini belirtmiştir.

Bu çalışmada PROMISE veri deposundan temin edilen, Albrecht, Finnish, Kemerer, Maxwell ve Miyazaki94 veri setleri kullanılmıştır. İlgili veri setleri, gerçek maliyet olan bağımlı ve maliyet ile ilgili olan bağımsız özniteliklerden oluşmaktadır. Ayyıldız (2007) çalışmasında veri setlerinde bulunan özniteliklerin yazılım maliyet tahminini çok etkilediğini vurgulamıştır. Bazı bağımsız özniteliklerin yazılım maliyetine fazla etkisi olmadığı, bu öz niteliklerin göz ardı edildiğinde daha iyi performans değerlerinin elde edilebileceği bu çalışma ile ispatlanmıştır.

Yazılım maliyet tahmininin doğruluğunu artırmak için birçok yazılım maliyet tahmin yöntemi geliştirilmiştir. Bu tahmin yöntemlerinden birisi de Yapay Zekâ yöntemleridir. Bu çalışmada yazılım projelerinin maliyeti, Yapay Zekâ teknolojisinin alt dallarından olan Makine Öğrenmesi ve özellikle GP kullanılarak tahmin edilmeye çalışılmıştır. Bu amaçla ilk olarak veri setlerine Genetik Programlama uygulanarak öznitelik seçimi yapılmıştır. Bu esnada maliyet tahminine etkisi olmayan ya da en az etkisi olan öznitelikler elenmiştir. Daha sonra veri setlerine WEKA'da bulunan Gaussian Processes (Gauss Yöntemi), Linear Regression (Doğrusal Regresyon), Multilayer Perceptron (Çok Katmanlı Algılayıcı), RandomSubSpace (Rastgele Alt Boşluk), Random Comittee (Rastgele Komite), Decision Table (Karar Tablosu), Random Tree (Rastgele Ağaç), Random Forest (Rastgele Orman) ve Genetic Programming Makine Öğrenmesi algoritmaları uygulanmıştır. Performans ölçütü olarak korelasyon katsayısı, MAE ve RAE baz alınarak değerlendirilmiştir. Testler WEKA 3.9 sürümü kullanılarak gerçekleştirilmiştir. GP algoritması için WEKA'nın 3.4.12 sürümü kullanılmıştır.

Bu çalışmaya GP ile yazılım maliyet tahmini konusunda öncü bir çalışma niteliği taşıdığından öncesinde bu işlemin doğru bir şekilde yapılıp yapılmayacağı araştırılmıştır. İncelenen bir çalışmada (Burgess ve Lefley, 2001) GP'nin doğrulukta önemli iyileştirmeler sağlayabileceği, ancak bu doğruluğun ölçülmesine ve yorumlanmasına bağlı olduğu belirtilmiştir. Ayrıca GP'nin yazılım maliyet tahmini için ek seçenek olma potansiyeline sahip olduğu bunun da kurulum ve çalıştırma çabasının yüksek ve yorumlanmasının zor olduğu vurgulanmıştır.

Shan vd. (2002) yaptıkları çalışmada, bazı araştırmacıların veri setlerinde, fazladan özniteliğin olduğunu ve bunları azaltmak için farklı yöntemleri uyguladıklarını vurgulamışlardır. İlgili çalışmada yazarlar, uygun metrik kümeleri belirlemeyi ve yazılım geliştirme çabasının tahminini iyileştirmeyi amaçlayan, geçmiş projelerin veri kümesine doğrusal olmayan modelleri uydurmak için evrimsel bir yaklaşım olan GGGP (Grammar Guided Genetic Programming – Dilbilgisi Güdümlü Genetik Programlama) kullanmışlardır. Bu şekilde yazılım maliyet tahmininde GP'yi başarılı bir şekilde kullanmışlardır.

Soleimanian vd. (2015) çalışmalarında GA ve Tabu Arama Algoritması'nın bir melez sistemini kullanarak yazılım projelerinin maliyet tahminini optimize etmeye çalışmışlar ve oluşturulan melez sistemle daha optimum sonuçların elde edildiğini göstermişlerdir.

Başkeleş vd. (2007) çalışmalarında yazılım maliyet tahmini için Makine Öğrenmesi algoritmalarını kullanan bir model önermişlerdir. Bu modeli kamuya açık veri setleri (NASA – National Aeronautics and Space Administration, USC – University of South California) ve Türkiye'deki yazılım kuruluşlarından toplanan veriler (SDR – Softlab Data Repository) üzerinde değerlendirmişlerdir. Değerlendirmeler ile bir veri kümesi için en iyi yöntemin değişebileceği ve tek bir modelin kullanılmasının her zaman en iyi sonuçları üretemeyeceği gerçeğini kanıtlamışlardır.

Singh ve Misra, (2012) çalışmasında, COCOMO (Constructive Cost Model – Yapıcı Maliyet Modeli)'nun daha iyi bir maliyet tahmini sağlayabilmesi için COCOMO modelinin parametreleri yeniden ayarlanmıştır. COCOMO modelinin parametrelerini ayarlamak için GP kullanılarak bir optimizasyon algoritması üzerinde çalışılmıştır. Geliştirilen modelin performansı NASA veri seti üzerinde test edilmiştir. Test sonuçları var olan modelle karşılaştırıldığında geliştirilen modelin daha iyi sonuçlar sağladığı tespit edilmiştir.

Tran vd., (2015) çalışması, yüksek boyutlu verilerde sınıflandırma, öznitelik sayısının fazla olması nedeniyle zorlu bir iş olduğu, sorunun GP ile özellik seçimi yapılarak çözülebileceği ve GP'nin hem özellik oluşturma hem de örtük özellik seçimi için kullanılabileceği sonucunu ortaya çıkarmıştır. Bu çalışmada yüksek boyutlu sınıflandırma problemlerinde öznitelik oluşturma ve seçimi için GP kullanımını araştıran kapsamlı bir çalışma sunmuşlardır. Öznitelik seçimi ve öznitelik oluşturma, özellik uzayının kalitesini artırmak için kullanılan veri ön işleme tekniklerdir. Öznitelik seçimi, orijinal öznitelik kümesinden yalnızca yararlı öznitelikleri seçmeyi amaçlar. Üç farklı türde öznitelik seçimi ve oluşturma yaklaşımı önerilmiştir: bunlar sarmalayıcı, filtre ve gömülü yaklaşımlar şeklindedir. Evrimsel hesaplama, özellik seçimi ve özellik oluşturma için yaygın olarak kullanılmaktadır. Bu makale, dört farklı sınıflandırma algoritması üzerinde, altı farklı şekilde seçilmiş öznitelik setinin performansını analiz ederek, yüksek boyutlu veriler üzerinde öznitelik inşası ve seçimi için GP'nin kullanımını araştırmaktadır.

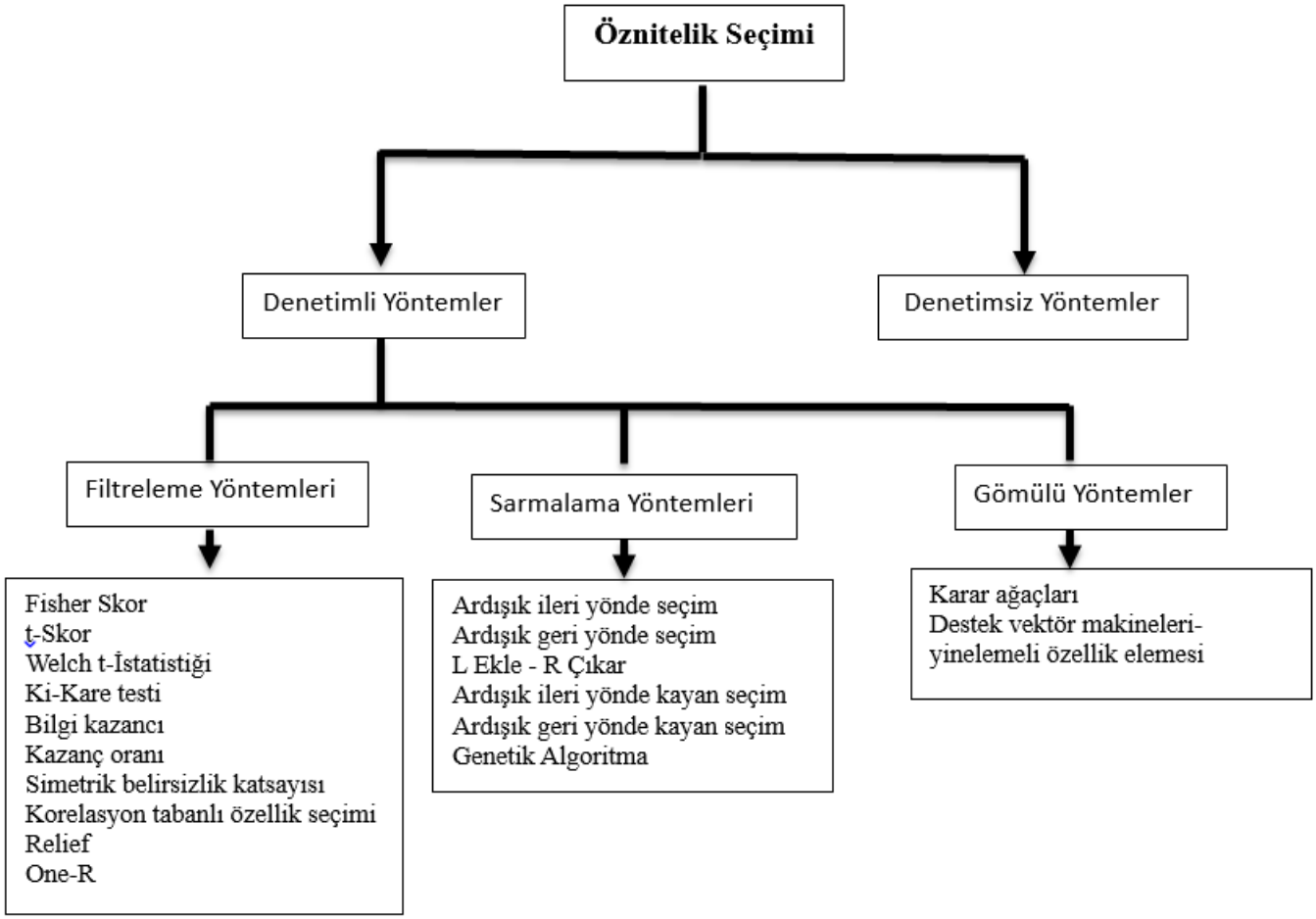
1.1. Öznitelik Seçimi

Makine Öğrenmesi'nde, bir modeli eğitmek ve daha iyi öğrenmesine yardımcı olmak için büyük miktarda veri toplanmaktadır. Çoğu zaman bu verilerin hepsi, sözkonusu model için yararlı olmamaktadır. Çok fazla gereksiz veriye sahip olmak modelin yavaşlamasına ve yanlış sonuçlar çıkarmasına neden olmaktadır. 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).

1.1.1. Öznitelik Seçim Yöntemleri

Öznitelik seçimi için kullanılan çeşitli algoritmalar mevcuttur. Bu algoritmalar, öğrenme algoritmasına bağımlılığına dayalı olarak filtreleme, sarmalama ve gömülü yöntemler olarak üç ana gruba ayrılır (Moghaddam, 2014). Filtreleme yöntemleri, veri madenciliğinde kullanılan en eski öznitelik seçim yöntemleri olarak bilinmektedir. Sadece istatistiksel ölçütlere dayalı fonksiyonlar yardımıyla öznitelik seçimi yapan yöntemlerdir. Sarmalama yöntemleri, öznitelikler üzerinde arama işlemi gerçekleştiren yöntemlerdir. Gömülü yöntemler ise, yapısında hem sınıflandırma algoritması hem de öznitellik seçimi algoritmasını barındırdığından, sınıflandırma ve öznitellik seçme süreçlerini eşzamanlı olarak gerçekleştirebilen yöntemlerdir (Budak, 2018). Şekil 1'de öznitelik seçim yöntemlerinden bazıları verilmiştir.



Şekil 1. Öznitelik seçimi yöntemleri

1.1.2. Öznitelik Seçim Algoritması

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 oluşturmak

yatmaktadır (Hall, 1999). CfsSubsetEval, herhangi bir açgözlü veya meta-sezgisel arama yaklaşımıyla kullanılabilir. Bu çalışmada CfsSubsetEval, GeneticSearch (Genetik Arama) algoritması ile birlikte kullanılmıştır.

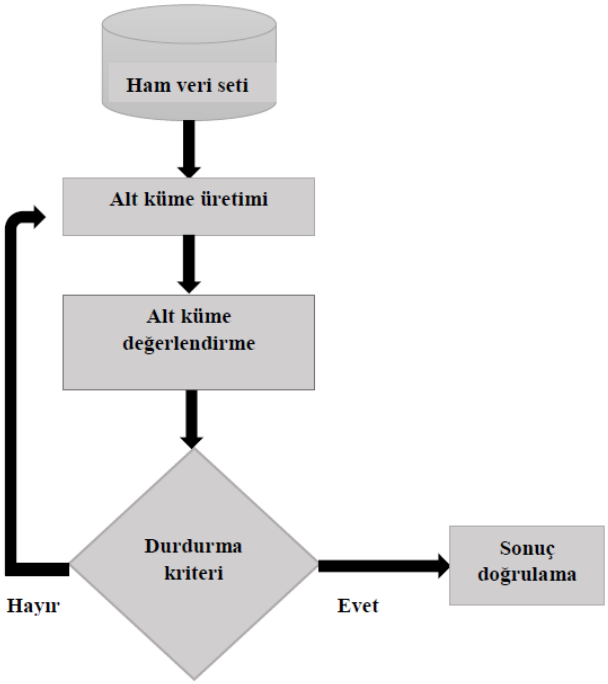
1.1.3. Öznitelik Seçimi Genel Adımları

Şekil 2'de görüldüğü üzere öznitelik alt kümesinin oluşturulması için ham veri setinden yararlanılmaktadır. Ham veriden elde edilen öznitelik alt kümesinin seçilip seçilmeyeceğine karar vermek için farklı formüller kullanılarak değerlendirmeler yapılmaktadır. Değerlendirmeler sonucunda seçilmesine karar verilen öznitelik, ilgili alt kümeye dâhil edilmekte ve algoritmanın durdurma kriteri sağlanana kadar süreç devam etmektedir.

1.2. Genetik Algoritmalar

GA evrimsel programlamanın bir alt dalıdır. Charles Darwin'in ilkelerine dayanan evrimsel programlama, 1960'lı yıllarda I.Rechenberg'in "Evrim stratejileri" adlı çalışmasıyla gündeme gelmiştir. Genetik Algoritmaların koda dökülüp programlanmasına GA'nın Genetik Programlama denmektedir. Genetik Algoritmalar, doğal seçim ilkesine dayanarak en iyi çözümü veya en iyiye yakın olabilecek çözümü bulmayı hedefleyen sezgisel bir arama ve optimizasyon yöntemidir (Nabiyev, 2016).

GA ilk defa Michigan Üniversitesi'nde John Holland tarafından 1975'te bugünkü biçimi ile kullanılmıştır. Holland evrim yasalarını GA içinde en iyileştirme problemleri için kullandığı çalışmaları bir araya getirmesi sonrasında GA, Yapay Zekâ ve özellikle Makine Öğrenmesi konularında büyük bir alt alan olarak kullanılmıştır. John Holland ve onun çalışma arkadaşları ile öğrencileri tarafından GA geliştirilmiş ve bilgisayar ortamına taşınmıştır. Daha sonra yine aynı ekipten ortaya çıkan ve gaz borularının GA ile optimizasyonunu inceleyen doktora tezi ile Genetik Algoritmaların sadece teorik olmadığı ve uygulamalarının farklı alanlarda yapılabileceği ispatlanmıştır. Bu tezin devamında gelen kitapta GA'ya dayalı tam 83 uygulamaya yer verilmiştir (Kubat, 2014; Moghaddam, 2014).



Şekil 2. Öznitelik seçimi akış şemas

GA'nın uygulama için en uygun olduğu problemler geleneksel yöntemler ile çözümü mümkün olmayan, matematiksel modeli kurulamayan ya da çözüm süresi problemin büyüklüğü ile üstel orantılı olarak artan problemlerdir. Bir arama algoritması olan GA, farklı bilim alanlarındaki optimizasyon problemlerini çözmede kullanılmaktadır (Kubat, 2014).

1.3. Makine Öğrenmesi

Yapay Zekânın bir alt dalı olan Makine Öğrenmesi, bilgisayarların insanlar gibi algoritmalar ve veriler yardımıyla öğrenmesini ve hareket etmesini sağlamaktadır (Kaluza, 2016). Makine Öğrenmesi'nin özünde, makinelerin tek başlarına doğru kararlar verebilme düşüncesi yatmaktadır. Bilgi güçtür prensibine dayanan Makine Öğrenmesi, bir makine ne kadar çok eğitilirse o kadar çok bilgilenir ne kadar çok bilgilenirse o kadar çok doğru kararlar verebilir. Yani bir makine eğitildikçe öğrenir, öğrendikçe daha doğru sonuçlar üretir (Ebren Kara ve Samli, 2021).

Gelecek ile ilgili tahminlerde bulunabilmek için geçmişteki verilerin analizinin yapılması gerekmektedir. Çok büyük miktardaki verinin elle işlenmesi ve analizinin yapılması çok mümkün değildir. Bu problemin çözümü için Makine Öğrenmesi algoritmaları geliştirilmiştir (Diri, 2014). Makine Öğrenmesi algoritmaları genellikle geçmiş verileri kullanarak sınıflandırma, öğrenme ve tahmin problemlerinde kullanılmaktadır (Mitchell, 1997).

1.3.1. Kullanılan Makine Öğrenmesi Algoritmaları

Gaussian Processes (Gauss Yöntemi): Gauss süreci, hem denetimli hem de denetimsiz öğrenme sürecinde olduğu gibi Bayes regresyonu için de kullanılabilen doğrusal olmayan güçlü

bir tahmin ve sınıflandırma algoritmasıdır. Gauss olasılık dağılımını genelleştiren parametrik olmayan stokastik bir süreçtir. Gauss süreci çok değişkenli Gauss dağılımlarını sonsuz boyutluluğa genişletir (Bishop, 2006).

Linear Regression (Doğrusal Regresyon): Doğrusal regresyon, iki değişken arasındaki sayısal ilişkiyi inceleyerek değişkenlerden birinin değerini başka bir değişkenin değerine göre kestirim yapmak için kullanılan bir analiz yöntemidir. Tahmin yapılmak istenen değişken, bağımlı değişken; bağımlı değişkenin değerini tahmin etmek için kullanılan değişkene de bağımsız değişken ismi verilmektedir.

Multilayer Perceptron (Çok Katmanlı Algılayıcı): Bir Yapay Sinir Ağı modeli olarak düşünülebilir. Belirli problemleri çözmek için birlikte çalışan, birbirine bağlı işlem elemanlarından (nöronlar veya düğümler) oluşan bir hesaplama sistemdir (Caudill, 1987). İnsan beyninin nasıl çalıştığından ilham alan, sinir sistemini modelleyerek oluşturulan bir algoritmadır. Çok Katmanlı Algılayıcı, giriş ve çıkış katmanı arasında bir veya daha fazla katman içeren ileri beslemeli bir sinir ağıdır. Temel olarak üç katman vardır: giriş katmanı, gizli katman ve çıkış katmanı. Gizli katman birden fazla olabilir. Her katmandaki her nöron (düğüm), bitişik katmanlardaki her nörona (düğüm) bağlıdır. Eğitim veya test vektörleri giriş katmanından verilir; gizli katmanda işlenir ve çıkış katmanından çıkış alınır (Gupta, 2015).

RandomSubSpace (Rastgele Alt Boşluk): Bu algoritma sınıflayıcılar topluluğunu oluşturmak için her bir giriş özniteliklerinin rastgele seçilmiş bir alt kümesini kullananan, karar ağacı tabanlı bir sınıflandırıcıdır. Algoritma öznitelik vektörününün alt kümelerinin boyutunu denetlemek için bir parametre üretmesinin yanında bir de tekrar sayısını ve kullanılacak rastgele adım sayısını sağlar.

Random Comittee (Rastgele Komite): Bu algoritma, temel sınıflandırıcılar topluluğunu rastgele hale getirecek şekilde oluşturur ve tahminlerini değerlendirir. Her bir sınıflayıcı aynı verilere fakat farklı rastgele sayı çekirdeğini kullanır. Tahmin sonucu her bir temel sınıflandırıcının ürettiği tahmin sonuçlarının bir ortalamasıdır.

Decision Table (Karar Tablosu): Bir karar tablosu sınıflandırıcısı oluşturan bir sınıflandırıcıdır. Karar tabloları tahmin için kullanılan sınıflandırma algoritmalarıdır. Bir karar tablosu, daha yüksek seviyeli bir tablodaki her girişin, başka bir tablo oluşturmak için bir çift ek özniteliğin değerlerine bölündüğü, hiyerarşik bir tablodan oluşur.

Random Tree (Rastgele Ağaç): Bu sınıflandırma algoritması, bütün düğümler için, belirli sayıdaki rastgele özellikleri dikkate alır ve bir ağaç oluşturur. Bu sınıflandırma algoritması budama yapmaz. Bunun yanında bir uzatma kümesine bağlı olarak sınıf olasılıklarının tahminine izin verme şansına da sahiptir.

Random Forest (Rastgele Orman): Rastgele ağaçlardan oluşan topluluklar kurarak rastgele ormanları kurar.

Genetic Programming (Genetik Programlama): Genetik Algoritmaların kodlanması ile oluşturulan programlardır. GP sınıflandırma için büyük bir potansiyel sunan; zor problemlerin çözümünde kullanılan evrimsel bir öğrenme tekniğidir. GP temsil biçimi olarak ağaç yapısını kullanır. Ağaç yapısında iç düğümler, işlevler ve operatörleri temsil ederken uç birimler değişkenleri ve sabitleri temsil ederler (Gupta, 2015).

2. Materyal ve Metot

2.1. Veri Setleri

Bu çalışmada yazılım maliyet tahmini için çok sık kullanılan Albrecht, Finnish, Kemerer, Maxwell ve Miyazaki94 veri setleri kullanılmıştır. Bu veri setleri PROMISE veri deposundan temin edilmiştir. Fakat bu veri setlerinden Finnish veri setine PROMISE veri deposundan artık erişilememektedir (Bosu ve Macdonell, 2019). Tablo 1'de kullanılan veri setlerine ait bilgiler verilmiştir.

Tablo 1. Veri setleri bilgileri.

Veri Seti	Kayıt Sayısı	Öznitelik Sayısı	Boyut (ölçü birimi)	Maliyet (ölçü birimi)
Albrecht	24	8	Fonksiyon Noktası	Adam-Saat
Finnish	38	9	Fonksiyon Noktası	Adam-Saat
Kemerer	15	8	KSLOC	Adam-Ay
Maxwell	62	27	Fonksiyon Noktası	Adam-Saat
Miyazaki94	48	9	KSLOC	Adam-Ay

Albrecht: Albrecht veri seti, IBM veri işleme hizmetlerinde gerçekleştirilen projelerden toplanan 24 kayıttan oluşur. Projeler COBOL, PL/I ve DMS programlama dilleri kullanılarak geliştirilmiştir. Projelerin boyutu ve karmaşıklığı, Albrecht tarafından önerilen fonksiyon noktası yaklaşımı kullanılarak ölçülmüştür (Albrecht ve Gaffney, 1983).

Finnish: Finnish veri seti, TIEKE organizasyonu tarafından Finlandiya'daki dokuz firmadan toplanmıştır. Projelerin boyutu ve karmaşıklığı fonksiyon noktası yaklaşımı kullanılarak ölçülmüştür. Gerçekte bu veri seti 40 kayıttan ve 9 öznitelikten oluşmaktadır fakat içinde eksik değerlerin olduğu verilerin kaldırılması ile 38 kayıt kalmıştır (Kitchenham ve Kansala, 1993).

Kemerer: Kemerer veri seti (Kemerer, 1987), veri işleme yazılımı geliştiren bir Amerikan firmasından toplanmıştır. Bu veri seti sekiz özniteliğe sahip 15 projeden oluşmaktadır. Veri setindeki en eski proje 1981'de başlamış olup projelerin çoğu 1983'te başlamıştır. Veri setindeki proje verileri 1985'te toplanmıştır. Projelerin binlerce kaynak kod satırına dayalı olarak orta ila büyük boyutta oldukları belirtilmiştir. Projelerin boyutu KSLOC (Kilo Source Lines of Code – Bin Kaynak Kod Satırı) olarak ölçülmüştür (Bosu ve MacDonell, 2019) .

Maxwell: Maxwell veri seti bir Fin ticarî bankasından toplanmıştır. 27 öznitelik ile temsil edilen 62 projeden oluşmaktadır (Maxwell, 2002). Projelerin başlangıç yılları 1985 ile 1993 yılları arasındadır.

Miyazaki94: Miyazaki94 veri seti, Fujitsu'nun Büyük Sistem Kullanıcıları Grubu tarafından toplanmıştır (Miyazaki vd., 1994). Veriler, 20 farklı kuruluşta ve bu kuruluşlardaki birden fazla bölümde geliştirilen 48 COBOL projesinden elde edilmiştir. Her proje için 9 öznitelik vardır.

2.2. Metot

Çalışmanın amacı, GP kullanılarak öznitelik seçiminin yazılım maliyet tahminine olan etkisinin araştırılmasıdır.

WEKA programı Yeni Zelanda'da bulunan Waikato Üniversitesindeki bir doktora öğrencisi tarafından geliştirilmiş olup elde edilen ham verilerin tanımlanması için kullanılmaktadır. WEKA programı, ham verileri sınıflandırma (classification), kümeleme (clustering), görselleştirme (visualize), bölütleme (segmentation), tahminleme (forecasting), veriler arasında ilişki kurma (associate), öznitelik seçimi (fuature selection), veri ön işleme (pre-processing) gibi Makine Öğrenmesi ve Veri Madenciliği işlemlerini gerçekleştirebilecek algoritmaları barındırmaktadır (Witten vd., 2011). Bu program ayrıca GPL (General Public Licence – Genel Kamu Lisansı) lisansına sahip ücretsiz olarak kullanılan açık kaynaklı bir uygulamadır.

Yazılım projelerinin maliyet tahmini için WEKA programı kullanılmıştır. İlgili veri setlerine WEKA'da bulunan Gaussian Processes, Linear Regression, Multilayer Perceptron, RandomSubSpace, Random Comittee, Decision Table, Random Tree, Random Forest ve Genetic Programming algoritmaları uygulanmıştır. Genetik Programming algoritması WEKA'nın 3.4.12 sürümünde çalıştırılmıştır. Ayrıca GeneticSearch algoritmasının Select attributes menüsünün altındaki Choose sekmesine eklenebilmesi için WEKA'nın ana penceresinde bulunan Tools menüsünden Package manager seçeneğinin tıklanarak GeneticSearch algoritmasının yüklenmesi gerçekleştirilmiştir.

Öznitelik seçiminin maliyet tahmine olan etkisini incelemek için geliştirilen metodoloji şu şekildedir: Veri setlerinde algoritmalar iki şekilde çalıştırılmıştır; ilkinde her hangi bir öznitelik seçimi yapılmadan ham veri seti üzerinden algoritmalar 10 kat çapraz doğrulama ile çalıştırılmış ve sonuçlar tablolaştırılmıştır; İkincisinde her bir veri setinde ilk önce Genetik Programlama uygulanarak öznitelik seçimi gerçekleştirilmiştir. Öznitelik seçimi sonrasında bazı öznitelikler veri setine dâhil edilmeyerek algoritmalar bu veri setleri üzerinde 10 kat çapraz doğrulama tekniği ile çalıştırılmış ve sonuçlar Tablo 2, Tablo 3, Tablo 4, Tablo 5 ve Tablo 6'da gösterilmiştir.

GP için farklı bir yöntem uygulanmıştır. Çünkü GP olasılıksal stokastik bir küresel arama algoritmasıdır; bu nedenle her popülasyonun her bireyinin, her yürütme sırasında arama alanı boyunca farklı bir yörünge gerçekleştirmesi ve popülasyonun çoklu (alt) optimal çözümlere yaklaşması beklenir. Bir GP'yi tekrar tekrar yürüterek bir dizi optimal çözüm toplanabilir. Bu çalışmada GP 15 defa çalıştırılarak, uygunluk değerine göre en uygun sonuç seçilmiştir. Bununla birlikte, ortalama değeri dikkate almak yanlıştır, çünkü benzer uygunluk değerine sahip iki alt optimal çözüm, tamamen farklı bir yapıya sahip olabilir, böylece ortalamaları arama uzayının uygun olmayan bir bölgesine karşılık gelebilir.

2.3. Performans Değerlendirmesi

2.3.1. Korelasyon Katsayısı (Correlation Coefficient)

Korelasyon, iki tesadüfi değişken arasındaki doğrusal ilişkinin gücünü ve yönünü belirtir. Korelasyon katsayısı negatif ise iki değişken arasında ters bir ilişki olduğu yani değişkenlerden biri artarken diğerinin azaldığı; korelasyon katsayısı pozitif ise değişkenler arasında doğrusal bir ilişki olduğu yani değişkenlerden biri artarken diğerinin de arttığı anlamına gelmektedir. Korelasyon katsayısının 0 olması ise iki değişken arasında herhangi bir ilişkinin olmadığını göstermektedir. Korelasyon katsayısı 1'e yaklaştıkça aradaki ilişkinin arttığı 0'a yaklaştıkça ise aradaki ilişkinin azaldığı anlaşılmaktadır.

2.3.2. MAE (Mean Absolute Error – Ortalama Mutlak Hata)

MAE, gerçek değer ile tahmin edilen değer arasındaki farkı bularak hata oranını hesaplamaktadır. Formülü Denk. (1)'de verilmiştir.

$$MAE = (|a1 - c1| + |a2 - c2| + \ldots + |an - cn|) / n \quad (1)$$

Burada c = tahmini değer, a = gerçek değer, n = örnek sayısı'dır.

2.3.3. RAE (Relative Absolute Error – Bağıl Mutlak Hata)

Gerçek değer ile hesaplama sonucu bulunan yaklaşık değer arasındaki farka mutlak hata denir. Bu farkların toplanıp gerçek değer ile gerçek değerin ortalaması arasındaki farkın toplamına bölünmesine bağıl mutlak hata denir. Formülü Denk. (2)'de verilmiştir.

$$Ej = \frac{\sum_{i=1}^{n} |P_{ij} - A_i|}{\sum_{i=1}^{n} |A_i - A_m|} \quad (2)$$

Burada $P_{ij} = i$ veri noktası için veri kümesi j tarafından tahmin edilen değer.
A_i = veri noktası için gerçek değer;
n = toplam veri noktası sayısı;
A_m = tüm A_i'lerin ortalaması

İdeal durumda, pay sıfıra eşittir ve *Ej* = 0'dır. Böylece *Ej* 0'dan sonsuza kadar değişir (Prabhakar ve Dutta, 2013).

3. Araştırma Sonuçları ve Tartışma

Yazılım maliyet tahmini için PROMISE veri deposundan temin edilen Albrecht, Finnish, Kemerer, Maxwell ve Miyazaki94 hazır veri setleri kullanılmıştır. Bu veri setlerine WEKA programında bulunan Makine Öğrenmesi algoritmaları farklı senaryolarda uygulanarak performans değerleri korelasyon katsayısı, MAE ve RAE baz alınarak Tablo 2, Tablo 3, Tablo 4, Tablo 5 ve Tablo 6'da gösterilmiştir.

Albrecht veri setine WEKA programının select attributes menüsü altındaki CfsSubsetEval ile GeneticSearch algoritması uygulanarak öznitelik seçimi yapılmıştır. Öznitelik seçimi yapılmış veri setinde öznitelik sayısı 8'den 3'e düşürülmüştür. CfsSubsetEval ile GeneticSearch uygulanarak seçilen öznitelikler Output, Inquiry, RawFPcounts olmuştur. Bağımlı öznitelik olan Effort özniteliği de özniteliklere eklenerek öznitelik sayısı 4 olarak belirlenmiştir. Algoritmalar Albrecht veri seti üzerinde öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra çalıştırılmış ve algoritmaların performans değerleri Tablo 2'de gösterilmiştir.

Tablo 2. Albrecht veri seti için öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra farklı algoritmalar ile elde edilen yazılım maliyeti tahmin sonuçlarının karşılaştırılması.

Albrecht veri seti						
	Genetik Algoritma ile öznitelik seçimi yapılmamış			**Genetik Algoritma ile öznitelik seçimi yapıldıktan sonra**		
ALGORİTMALAR	**Korelasyon katsayısı**	**MAE**	**RAE (%)**	**Korelasyon katsayısı**	**MAE**	**RAE (%)**
Gaussian Processes	0,652	15,873	77,850	0,884	21,719	106,520
Linear Regression	0,906	8,995	44,116	0,928	8,272	40,571
Multilayer Perceptron	0,754	12,055	59,126	0,935	7,2164	35,392
RandomSubSpace	0,433	14,314	70,201	0,667	12,724	62,403
Random Comittee	0,961	5,881	28,847	0,947	5,901	28,945
Decision Table	0,693	9,944	48,773	0,689	9,915	48,629
Random Tree	0,471	13,458	66,007	0,863	9,279	45,512
Random Forest	0,940	7,682	37,679	0,958	6,311	30,953
Genetic Programming	0,859	11,378	15,791	0,903	15,277	74,927

Tablo 2 incelendiğinde, Albrecht veri seti üzerinde öznitelik seçimi yapılmadan önce en iyi performansı 0,961 korelasyon katsayısı, 5,881 MAE ve % 28,847 RAE hata payı ile Random Comittee algoritması sergilerken öznitelik seçimi yapıldıktan sonra en iyi performansı 0,958 korelasyon katsayısı, 6,311 MAE ve % 30,953 RAE hata payı ile Random Forest algoritmasının gösterdiği görülmüştür. Veri setine uygulanan algoritmalar arasından RandomSubSpace algoritması öznitelik seçimi yapılmadan önce ve öznitelik seçimi yapıldıktan sonra düşük korelasyon katsayısı, yüksek MAE ve RAE hata payları ile en kötü performansı sergilemiştir.

Finnish veri setine CfsSubsetEval ile GeneticSearch algoritması uygulanarak öznitelik sayısı 9'dan 4'e düşürülmüştür. Seçilen öznitelikler dev.eff.hrs, FP, prod, lnsize olmuştur. Bağımlı öznitelik olan lneff özniteliği de özniteliklere eklenerek öznitelik sayısı 5 olarak belirlenmiştir. Algoritmalar, Finnish veri seti üzerinde öznitelik seçimi yapılmadan önce ve öznitelik seçimi yapıldıktan sonra çalıştırılmış ve algoritmaların performans değerleri Tablo 3'te gösterilmiştir. Tablo 3 incelendiğinde, hem öznitelik seçimi yapılmadan önce hem de öznitelik seçimi yapıldıktan sonra yüksek korelasyon katsayısı ve düşük MAE, RAE hata payları ile en iyi performansı Random Forest algoritması göstermiştir. Finnish veri setinde Genetic Programming algoritması çok kötü bir performans sergilemiştir. Veri setinde öznitelik seçimi yapılmadan önce algoritmanın korelasyon katsayısı 0,228, MAE 1,554 ve RAE % 151,881 iken öznitelik seçimi yapıldıktan sonra korelasyon katsayısı 0,395, MAE 1,322 ve RAE % 129,192 olduğu gözlemlenmiştir. Genetic Programming algoritmasının performans değerleri diğer algoritmaların performans değerleri ile beraber incelendiğinde hem öznitelik seçiminden önce hemde öznitelik seçiminden sonra düşük korelasyon, yüksek MAE ve RAE hata oranları ile kötü bir performans sergilediği gözlemlenmiştir.

Tablo 3. Finnish veri seti için öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra farklı algoritmalar ile elde edilen yazılım maliyeti tahmin sonuçlarının karşılaştırılması

Finnish veri seti						
	Genetik Algoritma ile öznitelik seçimi yapılmamış			**Genetik Algoritma ile öznitelik seçimi yapıldıktan sonra**		
ALGORİTMALAR	**Korelasyon katsayısı**	**MAE**	**RAE (%)**	**Korelasyon katsayısı**	**MAE**	**RAE (%)**
Gaussian Processes	0,859	0,569	55,616	0,8443	0,588	57,520
Linear Regression	0,960	0,254	24,826	0,960	0,254	24,826
Multilayer Perceptron	0,957	0,229	22,446	0,985	0,137	13,446
RandomSubSpace	0,923	0,375	36,670	0,945	0,294	28,779
Random Comittee	0,979	0,174	17,038	0,992	0,107	10,475
Decision Table	0,878	0,380	37,152	0,878	0,380	37,152
Random Tree	0,902	0,365	35,713	0,992	0,107	10,475
Random Forest	0,981	0,164	16,114	0,994	0,097	9,535
Genetic Programming	0,228	1,554	151,881	0,395	1,322	129,192

Kemerer veri setine CfsSubsetEval ile GeneticSearch algoritması uygulanarak öznitelik sayısı 8'den 4'e düşürülmüştür. Seçilen öznitelikler ID, Language, KSLOC, AdjFP olmuştur. Bağımlı öznitelik olan EffortMM özniteliği de özniteliklere eklenerek öznitelik sayısı 5 olarak belirlenmiştir. Algoritmalar, veri seti üzerinde öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra çalıştırılmış ve algoritmaların performans değerleri Tablo 4'te gösterilmiştir. Tablo 4 incelendiğinde, öznitelik seçimi yapılmadan önce ve öznitelik seçimi yapıldıktan sonra Genetic Programming algoritmasının diğer algoritmalara göre yüksek korelasyon katsayısı, düşük MAE ve RAE hata oranları ile en iyi performansı sergilediği gözlemlenmiştir. Kemerer veri setine uygulanan algoritmalar arasından Random Tree algoritmasının öznitelik seçimi yapılmadan önce -0,027 korelasyon katsayısı, 250,913 MAE ve % 155,911 RAE hata oranı ile en kötü performansı sergilediği; öznitelik seçimi yapıldıktan sonra en kötü performansı ise -0,037 korelasyon katsayısı,

148,196 MAE ve % 92,085 RAE hata payı ile RandomSubSpace algoritmasının sergilediği görülmüştür.

Maxwell veri setine CfsSubsetEval ile GeneticSearch algoritması uygulanarak öznitelik sayısı 27'den 19'a düşürülmüştür. Seçilen öznitelikler Syear, App, Har, Dba, Source, T01, T02, T04, T06, T07, T08, T09, T10, T11, T13, T14, Duration, Size, Time olmuştur. Bağımlı öznitelik olan Effort özniteliği de özniteliklere eklenerek öznitelik sayısı 20 olarak belirlenmiştir. Algoritmalar, veri seti üzerinde öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra çalıştırılmış ve algoritmaların performans değerleri Tablo 5'te gösterilmiştir. Tablo 5 incelendiğinde, öznitelik seçimi yapılmadan önce ve öznitelik seçimi yapıldıktan sonra Linear Regression algoritmasının diğer algoritmalara göre yüksek korelasyon katsayısı, düşük MAE ve RAE hata oranları ile en iyi performansı sergilediği gözlemlenmiştir. En kötü performansı ise düşük korelasyon katsayısı, yüksek MAE ve RAE hata oranları ile Decision Table algoritmasının gösterdiği görülmüştür.

Miyazaki94 veri setine CfsSubsetEval ile GeneticSearch algoritması uygulanarak öznitelik sayısı 9'dan 3'e düşürülmüştür. Seçilen öznitelikler KLOC, FORM, FILE olmuştur. Bağımlı öznitelik olan MM özniteliği de özniteliklere eklenerek öznitelik sayısı 4 olarak belirlenmiştir. Algoritmalar, veri seti üzerinde öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra çalıştırılmıştır ve algoritmaların performans değerleri Tablo 6'da gösterilmiştir. Tablo 6 incelendiğinde, öznitelik seçimi yapılmadan önce en iyi performansı 0,757 korelasyon katsayısı, 38,001 MAE ve % 101,039 RAE hata oranı ile Genetic Programming algoritması göstermiştir. Öznitelik seçimi yapıldıktan sonra ise en iyi performansı 0,783 korelasyon katsayısı, 21,977 MAE ve % 58,435 RAE hata oranı ile Random Forest algoritmasının sunduğu görülmüştür. Veri setine uygulanan Decision Table algoritması hem öznitelik seçiminden önce hemde öznitelik seçiminden sonra 0,131 korelasyon katsayısıyla en düşük performansı sergilemiştir.

Tablo 4. Kemerer veri seti için öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra farklı algoritmalar ile elde edilen yazılım maliyeti tahmin sonuçlarının karşılaştırılması.

Kemerer veri seti						
	Genetik Algoritma ile öznitelik seçimi yapılmamış			**Genetik Algoritma ile öznitelik seçimi yapıldıktan sonra**		
ALGORİTMALAR	**Korelasyon katsayısı**	**MAE**	**RAE (%)**	**Korelasyon katsayısı**	**MAE**	**RAE (%)**
Gaussian Processes	0,240	173,476	107,793	0,270	161,166	100,145
Linear Regression	0,369	173,240	107,647	0,342	190,216	118,195
Multilayer Perceptron	0,351	129,458	80,442	0,327	150,462	93,493
RandomSubSpace	0,024	144,782	89,963	-0,037	148,196	92,085
Random Comittee	0,325	142,710	88,676	0,186	169,009	105,018
Decision Table	0,102	144,995	90,096	0,302	176,223	109,500
Random Tree	-0,027	250,913	155,911	0,329	163,931	101,862
Random Forest	0,353	129,056	80,192	0,292	143,935	89,437
Genetic Programming	0,529	207,073	128,670	0,650	140,451	87,273

Tablo 5. Maxwell veri seti için öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra farklı algoritmalar ile elde edilen yazılım maliyeti tahmin sonuçlarının karşılaştırılması.

Maxwell veri seti						
	Genetik Algoritma ile öznitelik seçimi yapılmamış			**Genetik Algoritma ile öznitelik seçimi yapıldıktan sonra**		
ALGORİTMALAR	**Korelasyon katsayısı**	**MAE**	**RAE (%)**	**Korelasyon katsayısı**	**MAE**	**RAE (%)**
Gaussian Processes	0.783	3925,049	62,473	0,787	3933,870	62,613
Linear Regression	0,808	4157,589	66,174	0,854	3395,066	54,037
Multilayer Perceptron	0,764	4764,378	75,832	0,816	4146,309	65,995
RandomSubSpace	0,669	4734,964	75,364	0,647	4700,819	74,821
Random Comittee	0,787	3991,499	63,531	0,692	4238,346	67,460
Decision Table	0,313	5355,558	85,242	0,417	5060,946	80,553
Random Tree	0,569	5686,967	90,517	0,589	5211,558	82,950
Random Forest	0,761	3998,217	63,638	0,762	3827,568	60,921
Genetic Programming	0,618	7700,821	122,570	0,450	8906,493	141,761

Tablo 6. Miyazaki94 veri seti için öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra farklı algoritmalar ile elde edilen yazılım maliyeti tahmin sonuçlarının karşılaştırılması.

Miyazaki94 veri seti						
	Genetik Algoritma ile öznitelik seçimi yapılmamış			**Genetik Algoritma ile öznitelik seçimi yapıldıktan sonra**		
ALGORİTMALAR	**Korelasyon katsayısı**	**MAE**	**RAE (%)**	**Korelasyon katsayısı**	**MAE**	**RAE (%)**
Gaussian Processes	0,523	40,605	107,963	0,625	40,133	106,709
Linear Regression	0,292	35,355	94,005	0,752	25,096	66,726
Multilayer Perceptron	0,704	23,149	61,551	0,519	39,596	105,280
RandomSubSpace	0,478	32,952	87,616	0,717	25,341	67,380
Random Comittee	0,605	28,591	76,021	0,702	26,861	71,421
Decision Table	0,131	36,197	96,243	0,131	35,032	93,146
Random Tree	0,181	40,931	108,830	0,612	32,005	85,096
Random Forest	0,674	29,924	79,565	0,783	21,977	58,435
Genetic Programming	0,757	38,001	101,039	0,629	46,798	124,430

4. Sonuç

Bu çalışmanın amacı PROMISE veri deposundan alınan eski yazılım proje verilerinin tutulduğu Albrecht, Finnish, Kemerer, Maxwell ve Miyazaki94 veri setlerinde Genetik Algoritma kullanırak öznitelik seçiminin yapılması ve öznitelik seçiminin yazılım maliyet tahminine olan etkisinin araştırılmasıdır.

Bu çalışmada yazılım maliyet tahmini için Makine Öğrenmesi algoritmaları test edilmiştir. Bunun için WEKA aracında bulunan Gaussian Processes, Linear Regression, Multilayer Perceptron, RandomSubSpace, Random Comittee, Decision Table, Random Tree, Random Forest ve Genetic Programming algoritmaları kullanılmıştır. Algoritmalar ilgili veri setleri için iki şekilde çalıştırılmıştır. İlk önce ham veri seti üzerinde çalıştırılan algoritmalar, daha sonra veri setlerine öznitelik seçimi yapılarak tekrar çalıştırılmıştır. Algoritmaların performans değerleri Tablo 2, Tablo 3, Tablo 4, Tablo 5 ve Tablo 6 da gösterilmiştir. Tablolar incelendiğinde Genetik Algoritma kullanılarak veri setleri üzerinde öznitelik seçiminin yapılması Makine Öğrenmesi algoritmalarının performans değerlerini dikkat çekici şekilde iyileştirmiştir. Bu çalışmada ayrıca WEKA'nın eski sürümünde bulunan Genetic Programming algoritması da yazılım maliyet tahmini için kullanılmıştır. Analiz sonuçları incelendiğinde Genetic Programming algoritmasının yazılım maliyet tahmininde başarılı bir şekilde kullanılabilindiği gözlemlenmiştir.

Tablo 2'de performans değerlerine bakıldığında, en iyi performansın 0,961 korelasyon katsayısı ve 5,881 MAE hata payı ile Random Comittee algoritması tarafından elde edildiği görülmüştür. Veri seti üzerinde öznitelik seçimi yapıldıktan sonra en iyi tahmin sonucunu 0,958 korelasyon katsayısı ve 6,311 MAE hata payı ile Random Forest algoritması bulmuştur. Albrecht veri setinde 0,433 korelasyon katsayısı ve 14,314 MAE hata payı ile en kötü tahmini yapan RandomSubSpace algoritmasının öznitelik seçiminden sonra korelasyon katsayı 0,667'ye yükselmiş ve MAE hata payı da 12,724'e düşmüştür.

Finnish veri setinde en iyi tahmin sonucunu 0,981 korelasyon katsayısı ve 0,164 MAE hata payı ile Random Forest algoritması bulmuştur. Veri seti üzerinde öznitelik seçimi yapıldıktan sonra korelasyon katsayısı 0,994'e yükselmiş ve MAE hata payı da 0,097'ye düşmüştür. Finnish veri setinde 0,228 korelasyon katsayısı ve 1,554 MAE hata payı ile en kötü tahmini yapan Genetic Programming algoritması olmuştur. Veri seti üzerinde öznitelik seçimi yapıldıktan sonra algoritmanın korelasyon katsayı 0,395'e yükselmiş, MAE hata payı 1,322'ye düşmüştür.

Kemerer veri setine uygulanan Makine Öğrenmesi algoritmaları incelendiğinde en iyi performansı 0,529 korelasyon katsayısı ve 207,073 MAE hata payı ile Genetic Programming algoritması göstermiştir. Öznitelik seçimi yapıldıktan sonra korelasyon katsayı 0,650'a yükselmiş, MAE hata payı 140,451'e düşmüştür. Kemerer veri setinde en kötü tahmin sonucunu öznitelik seçiminden önce -0,027 korelasyon katsayısı ve 250,913 MAE hata payı ile Random Tree algoritması, öznitelik seçiminden sonra -0,037 korelasyon katsayısı ve 148,196 MAE hata payı ile RandomSubSpace algoritması bulmuştur.

Linear Regression algoritması Maxwell veri setinde yazılım maliyet tahmini için en iyi performansı göstermiştir. Veri setinde iki şekilde çalıştırılan algoritma ham veri seti üzerinde 0,808 korelasyona katsayısı ile çalışırken öznitelik seçimi yapıldıktan sonra korelasyon katsayısı 0,854'e yükselmiştir. En kötü tahmin sonucunu bulan algoritma 0,313 korelasyon katsayısı ile Decision Table algoritması olmuştur. Veri seti üzerinde öznitelik seçimi yapıldıktan sonra algoritmanın korelasyon katsayısı 0,417'ye yükselmiştir.

Miyazaki94 veri setine yazılım maliyet tahmini için uygulanan Makine Öğrenmesi algoritmalarının performans değerleri incelendiğinde en iyi tahmini gerçekleştiren 0,757 korelasyon katsayısı ve 38,001 MAE hata payı ile Genetic Programming algoritması olmuştur. Veri seti üzerinde öznitelik seçimi yapıldıktan sonra en iyi performansı gösteren algoritma 0,783 korelasyon katsayı ve 21,977 MAE hata payı ile Random Forest algoritması olmuştur. 0,131 korelasyon katsayısı ile Decision Table algoritması veri setine uygulanan öznitelik seçiminden önce ve öznitelik seçiminden sonra en kötü performansı sergilemiştir.

Yazılım projelerinin maliyet tahmini için kullanılan yöntemlerden herhangi birinin diğerinden daha üstün olduğunu söylemek çok doğru bir yaklaşım değildir (Kumari ve Pushkar, 2013). Bu çalışma ile yazılım maliyet tahmini için kullanılan Makine Öğrenmesi algorimalarından herhangi birisinin her zaman en iyi sonucu üretmediği ispatlanmıştır. Yazılım maliyet tahmini için kullanılan Makine Öğrenmesi algoritmaları uygulandıkları veri setlerine göre performans değerleri değişmektedir. Bu çalışmanın sonucunda elde edilen performans

değerleri incelendiğinde Genetic Programming algoritmasının, kullanılan diğer Makine Öğrenmesi algoritmalarına göre Kemerer veri seti üzerinde en iyi performansı segilerken Finnish veri seti üzerinde en kötü performansı sergilediği gözlemlenmiştir.

Yazılım projelerinin maliyet tahmini için oldukça fazla literatür çalışması mevcuttur. Bu çalışmaların çoğunda kullanılan hazır veri setleri üzerinde herhangi bir öznitelik seçimi yapılmadan yazılım maliyet tahmini gerçekleştirilmiştir. Bu çalışmada veri setlerine WEKA programının select attributes menüsü altındaki CfsSubsetEval ile GeneticSearch algoritması uygulanarak öznitelik seçimi yapılmıştır. Kullanılan veri setleri üzerinde öznitelik seçimi yapılmadan ve öznitelik seçimi yapıldıktan sonra yazılım maliyet tahmini gerçekleştirilmiştir. Yazılım projelerinin maliyet tahmini için kullanılan hazır veri setleri üzerinde öznitelik seçimi yapılarak yazılım maliyet tahmininin gerçekleştirilmesi genel olarak tahmin sonuçlarının doğruluk oranlarını artırmıştır. Bu çalışma sayesinde öznitelik seçiminin yazılım maliyet tahminine olan etkileri incelenmiş ve bir veri deposundan temin edilen hazır veri setlerinin kullanılmadan önce öznitelik seçiminin yapılıp o şekilde ilgili algoritmalarda kullanılmasının doğruluk oranlarını artıracağı bilgisine ulaşılmıştır.

Gelecek çalışmalarda güncel yazılım projelerinin veri setleri üzerinde faklı öznitelik seçimi yöntemleri kullanılarak yazılım maliyet tahmini gerçekleştirilmesi planlanmaktadır.

Kaynakça

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.*

Albrecht, A. J., Gaffney. J. E., 1983, *Software function, source lines of code, and development effort prediction: a software science validation.* IEEE Trans. Softw. Eng. *9, 6 (1983), 639*.

Ayyıldız, M., 2007, *Yazılım Projeleri Ölçüm Sonuçları Veri Tabanının Oluşturulması ve Yeni Yazılım Projelerinin Maliyet Tahmininde Kullanımı*, Doktora Tezi, Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü.

Başkeleş, B., Turhan, B., Bener, A., 2007, *Software Effort Estimation Using Machine Learning Methods,* Computer and information sciences, Ankara, IEEE.

Bishop, C. M., 2006, *Pattern recognition and machine learning*, Springer, New York.

Bosu, M.F., Macdonell, S.G., 2019, *Experience: Quality Benchmarking of Datasets Used in Software Effort Estimation*, ACM Journal of Data and Information Quality, *11(4), 1 – 38*.

Budak, H., 2018, *Özellik Seçim Yöntemleri ve Yeni Bir Yaklaşım*, Süleyman Demirel Üniversitesi Dergisi.

Burgess, C.J., Lefley, M., 2001, *Can Genetic Programming Improve Software Effort Estimation? A Comparative Evaluation*, Information and Software Technology, *43, 863.*

Caudill, M., 1987, *Neural networks primer*, J. AI Expert, *2(12), 46 – 52.*

Demirörs, O., 2011, *Yazılım Kestirimi İçin Referans Veri Kümesi Ve Süreç Odaklı Bir Yöntem*, 5. Ulusal Yazılım Projeleri Sempozyumu-UYMS.

Diri, B., 2014, *Makine Öğrenmesine Giriş*, Ders Notları, https://www.siskon.com.tr/dosya/PDF/Makale/Makina_Ogrenmesi.pdf, [Ziyaret Tarihi: 24.06.2021].

Ebren Kara, Ş., Şamlı, R., 2021, *Yazılım Projelerinin Maliyet Tahmini için WEKA'da Makine Öğrenmesi Algoritmalarının Karşılaştırmalı Analizi*, Avrupa Bilim ve Araştırma Dergisi, *23, 415 – 426.*

Gupta, A., 2015, *Classification Of Complex UCI Datasets Using Machine Learning And Evolutionary Algorithms*, International Journal Of Scientific & Technology Research, *4(5), 85 – 94.*

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

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

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.*

Kaluza, B., 2016, *Machine Learning in Java*, Pact Publishing.

Kemerer, C.F., 1987, *An Empirical Validation Of Software Cost Estimation Models. Commun*. ACM 30, 5(1987), 416–429.

Kitchenham B., Kansala. K., 1993, *Inter-item correlations among function points.* International Conference on Software Engineering. *229 – 238.*

Kubat, C., 2014, *Matlab Yapay Zeka ve Mühendislik Uygulamaları*, 2. Baskı, Pusula Yayınları, İstanbul, ISBN: 978-605-5106-12 – 6.

Kumari, S. ve Pushkar, S. 2013, Performance Analysis of the Software Cost Estimation Methods: A Review, International Journal of Advanced Research in Computer Science and Software Engineering, *229 –238.*

Maxwell, K., 2002, *Applied Statistics for Software Managers*, Prentice-Hall, Englewood Cliffs.

Mitchell, T. M., 1997, *Machine Learning*, McGraw-Hill and MIT Press.

Miyazaki, Y., Terakado, M., Ozaki,K., Nozaki. H., 1994, *Robust Regression For Developing Software Estimation Models*. J. Syst. Softw. *27, 13 – 16.*

Moghaddam, S.A.V., 2014, *Etkin Sınıflandırma İçin Genetik Algoritma Tabanlı Öznitelik Alt Küme Seçimi*, Yüksek Lisans Tezi, Gazi Üniversitesi, Fen Bilimleri Enstitüsü.

Nabiyev, V. V., 2016, *Yapay Zeka*, 5. Baskı, Seçkin Yayınları, Ankara, ISBN: 978-975-02-3727-0.

Prabhakar, Dutta, M., 2013, *Application Of Machine Learning Techniques For Predicting Software Effort*, Elixir Comp. Sci. & Engg., *56, 13677 – 13682.*

Shan, Y., *McKay*, C.J., Essam, D.L., 2002, *Software Project Effort Estimation Using Genetic Programming*, International Conference on Communications Circuits and Systems.

Singh B.K., Misra, A.K., 2012, *Software Effort Estimation by Genetic Algorithm Tuned Parameters of Modified Constructive Cost Model for NASA Software Projects*, International Journal of Computer Applications, *59(9).*

Soleimanian, F., Rezaii, R., Arasteh, B., 2015, *A New Approach by Using Tabu Search and Genetic Algorithms in Software Cost Estimation,* International Conference on Application of Information and Communication Technologies.

Tran, B., Xue, B., Zhang, M., 2015, *Genetic programming for feature construction and selection in classification on high-dimensional data*, Springer-Verlag Berlin Heidelberg: Regular Papers, *8, 3–15.*

Vikipedi, 2020, *Korelasyon,* Korelasyon – Vikipedi https://tr.wikipedia.org/wiki/Korelasyon.

Yücalar, F., 2011, *Use-Case Tabanlı Yazılım Emek Kestirim Modeli*, Doktora Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü.