# Yazılım Geliştirme Projelerinde Yapay Sinir Ağı Kullanarak Maliyet Tahmini

Murat AYYILDIZ[1] , Oya KALIPSIZ[1] ve Sırma YAVUZ[3]

[1,2,3]Bilgisayar Mühendisliği Bölümü, Yıldız Teknik Üniversitesi, İstanbul
Beşiktaş, İstanbul. *e-posta: {f0100301, kalipsiz, sirma}@ce.yildiz.edu.tr*

## Özetçe

Yazılım maliyet tahminleme çalışmalarında ölçüt kümesi seçiminin hayati bir rolü vardır; bunun önemi özellikle yapay sinir ağı bazlı çalışmalarında göz ardı edilmiştir. Bu çalışmada yeni bir ölçüt kümesi oluşturduk, oluşturduğumuz ölçüt kümesi için gerçek hayattan projelerden veriler toplandık. Ölçüt kümesini kullanarak yeni bir yapay sinir ağı modelleri oluşturduk. Elde edilen sonuçları geleneksel ölçütleri kullanarak yapılan önceki çalışma sonuçlarıyla karşılaştırdık. Karşılaştırmayı yapabilmek için iki tip veri kullanılmıştır. Birinci kısım önceki çalışmalarda genellikle kullanılan yapıcı maliyet modeli'nden (COCOMO) alındı ve ikinci kısım ise yeni oluşturulan ölçüt kümesine göre Türkiye'de bulunan uluslararası bir firmadan toplanan veriler kullanıldı. Burada sunulan model MLP ve Elman bazlı olup her ikisi içinde COCOMO 81 ve yeni oluşturulan ölçüt kümesi kullanılmaktadır.
Bu çalışmada maliyet tahmini için her iki yaklaşımın performansını da karşılaştırdık. Doğru olarak karşılaştırabilmek için veri kümelerinde eşit sayıda örnek ile test ettik. Daha iyi araştırma adına yeni oluşturulan veri kümesi için daha büyük bir küme ile de çalışmalar yaptık. Ölçüt kümelerinin karakteristiği ve veri miktarı bu çalışmanın incelediği konulardan biridir.
Yazılım maliyet tahminleme çalışmalarında olan bir diğer zorluk veri toplamanın zaman ve dikkat gerektirdiği gerçeğidir. Toplanan verilerin daha bütünsel kullanımı için k-fold çapraz onaylama metodu kullanılmıştır. Toplanan veri limitli olması nedeniyle %5,%10 ve %15'lik çapraz onaylama teknikleri de uygulanmıştır.
Doğru ve nitelikli bir ölçüt kümesi kullanıldığı sürece yapay sinir ağları yazılım maliyet tahminleme çalışmalarında başarıyla kullanılabilinir.

## 1. Giriş

Bilgisayar yazılımlarında maliyet tahmini son zamanlarda oldukça önem kazanmıştır. Çünkü yazılım geliştirme maliyeti her geçen gün artmakta ve yazılım geliştirmek için daha fazla harcamalar yapılmaktadır. Yazılım maliyet tahmini hem devletler için hem de organizasyonlar için önemli bir problemdir. Planlanan zaman ve bütçeyi aşan oldukça çok sayıda proje mevcuttur. Bu aşmaların çoğunun temelinde baştan bütçe ve zaman tahminini doğru yapamamaktan kaynaklanan başarısızlıklar yatmaktadır.

"Yazılım mühendisliği" bir yazılım ürünü inşa etmek için kullanılan teknikler topluluğunu tanımlamak için kullanılan bir terimdir. Burada "mühendislik" yaklaşımıyla yönetim, bütçeleme, planlama, modelleme, tasarlama, uygulama, test etme ve bakım konularını içine almaktadır. Günümüzde yazılım projeleri geliştirirken ölçülebilir hedefler koymakta başarılı olunamamaktadır. Örneğin bir yazılıma başlarken onun "güvenilir", "bakım yapılabilir", "kullanıcı dostu" gibi olacağı söylenmekte, net tanımlar verilmemektedir. Ölçme işinin açık ve net hedefleri olmalıdır. Yazılım mühendisliğinde ölçütlerin kullanımı teorik ve pratik çalışmalar arasındaki aralığı küçültecek bir anahtardır. Ölçme ve değerlendirme yapabilmek için sistematik ölçüm yapmak gerekmektedir.

Bu çalışmada öncelikli olarak belirlenmiş yöntemlere göre yazılım maliyetleme çalışmalarında kullanılabilinecek yeni bir ölçüt kümesi oluşturmuş ve sonra bir çok yapay sinir ağı modeli incelemesinden sonra bir MLP ve Elman kullanarak bir model ortaya koymuştur.

## 2. Oluşturulan Ölçüt Kümesi

### 2.1 Ölçüt Kümesi Belirleme Metodolojisi

Ölçüt terimi genellikle belirli bir öge veya bir süreç için yapılan belirli ölçümler kümesini ifade etmek için kullanılır. Yazılım Mühendisliği ölçütleri ölçüm birimleridir.
Bunların temelde ölçtükleri:

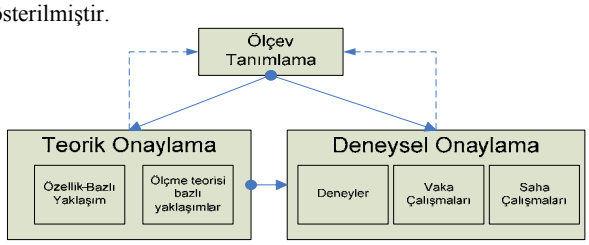
1. Yazılım Mühendisliği ürünleri (Tasarım, kaynak kodu, test vakaları)
2. Yazılım Mühendisliği süreçleri (Analiz, tasarım, kodlama aktiviteleri)
3. Yazılım Mühendisliği kişileri (Test eden kişilerin üretkenliği ve verimi)

Dolayısıyla yazılım mühendisliği için oluşturulacak ölçüt kümelerinde ürün, süreç ve kişiler temel yapı taşlarıdır şeklinde düşünülebilinir.

Ölçüt bir sistemin veya bir parçanın verilen bir özelliğinin nicel ölçüm derecesidir. Ölçüt hesaplanabilir veya birleşik bir gösterge olabilir. Bir yazılım ihtiyacının büyüklüğünün ölçülebileceği fikrini ilk ortaya çıkaran ve bir sistem kuran Allan Albrecht'tir. 1979'da IBM'de çalışan Albrecht işlev puan analizi adıyla bir metod ortaya koymuştur. California'da bir danışmanlık firması olan TRW'de bulunan bir çok proje üzerinde çalışarak, Boehm COCOMO'yu ortaya koymuştur. COCOMO göreceli olarak dosdoğru bir modeldir. Yapay sinir ağı kullanarak yazılım maliyet tahminleme konusunda yapılmış olan birçok çalışmada COCOMO '81 ölçüt kümesini kullanmıştır. Ancak kullanılabilinecek seviyede iyi sonuçlar alınılmamıştır. Burada kritik nokta ölçüt setinin oluşturulmasına yeteri kadar önem verilmemesidir. Bu yüzden biz çalışmadan önce bir ölçüt seti oluşturmaya karar verdik. Oluşturduğumuz bu ölçüt setine YEEM (Yıldız Effort Estimation Metrics) ismini verdik. Birçok çalışmada COCOMO '81 ölçüt kümesini kullanması dolayısıyla oluşturduğumuz ölçüt kümesinde COCOMO '81'i içermesi bize çalışma sonuçlarını bir nebze karşılaştırabilme şansı tanıyacağının önemini ihmal etmedik. Oluşturuluan ölçüt kümesi temel olarak COCOMO 81, COCOMO II, 21 adet gerçek yazılım projesini, toplam 90 yıllık yazılım geliştirme tecrübesi olan 28 yazılım geliştirme proje yöneticisi öneri ve fikirleri üzerinde çalışılarak hazırlanmıştır.

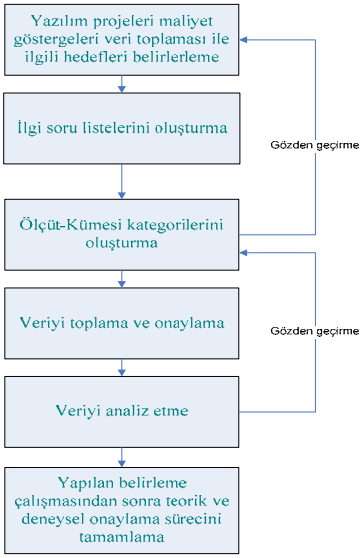
Literatürde ölçüt belirleme konusunda oldukça az çalışma vardır. Yazılım mühendisliği ölçütlerini belirlemek oldukça zordur. İyi tanımlanmamış hedefler, iyi tanımlanmamış hipotezler ve gözlemsel onaylama süreçleri eksikligi temel nelenleridir [1]. Ölçüt belirleme çalışması metodolojik olarak tanımlanmalıdır. Şekil 1'de

kullandığımız ölçüt belirleme çalışmasının temel görevleri gösterilmiştir.




Şekil.1 : Yazılım ölçütlerinin tanımlama ve onaylama metodu

Biz net ölçüm hedefleri üzerinde bir ölçüt seti oluşturmaya çalıştık. Bu yüzden hedefe yönelik ölçüm (GQM : Goal Oriented Measure) yaklaşımını ölçüt seçme ve onaylama süreci için kullandık. Basili ve Weiss [1] veri toplama metodolojisini YEEM'I ("Yıldız Effort Estimation Metrics") oluşturmak için kullandık Bu metodolojinin geribeslemeli 6 adımı (Şekil 2) vardır.




Şekil.2 : Yazılım ölçütlerinin tanımlama ve onaylama metodu

Yapılan belirleme çalışmasından sonra teorik ve deneysel onaylama (validation) sürecini tamamlamız gerekti. Teorik onaylama süreci temelde iki türlüdür:

a. Özellik bazlı yaklaşımlar ([3], [1])
b. Ölçüm teorisi bazlı yaklaşımlar ([4], [5], [7])

Bu çalımada YEEM'in teorik onaylama süreci için "distance framework" modelini kullandık. Bu model Poels ve Dedene [6] tarafından ortaya konmuş ölçme teorisi bazlı yazılım ölçüt onaylama için kavramsal bir çatıdır. Bu çatı bu bildirinin kapsamı dışındadır. Detayları kaynaklar [6] kısmında bulabilirsiniz. Bu çatının 5 aktivitesi vardır. Bunlar (i)Bir ölçüm soyutlaması bulunur, (ii) Uzaklık tanımlanır, (iii) Ölçüm soyutlamaları arasında ki uzaklıklar nitelendirilir (iv) Referans bir soyutlama bul, (v) Bu özellik için bir ölçme tanımla. Deneysel onalaylama süreci bu tür çalışmalar aslıda gerçek bir kanıttır. YEEM ölçüt kümesi için saha çalışmalar ve vaka çalışmaları yapılmıştır.

### 2.2 YEEM ölçüt kümesi

Ölçüt belirleme çalışmamızda dikkat ettiğimiz temel noktalar şunlardır:

- Ölçütler ayrık, yalıtılmış olmalıdır.
- Süreç, kişi, ürün bazında mümkün olduğunca çok ölçüt bulup daha sonra yapılacak bir analiz ile en yararlıların bulunmasının sağlanmalıdır.
- Ölçütlerin birbirleriyle ilişkili olmamaması gerekmektedir. Birbirleri üzerinde etkisi olmamalıdır veya çok az olmalıdır. Bu sürekli izlenmelidir.
- Ölçüt bulma, çıkarma süreci sistematik ve düzenli olmalıdır. Sürekli iyileştirme yapmak gerekmektedir.
- Ölçütlerin doğruluğu gerçek hayat ile kontrol edilebilmelidir.
- İstatistiksel yöntemlele ölçütlerin hata analizleri yapılmalıdır. Böylece gereksiz hatta zararlı ölçütler temizlenmelidir.
- Ölçüt bazında mantıklı ve doğru karşılaştırmalar yapabilmek için kişi, süreç ve ürün benzerlikleri ve farklılıkları karşılaştırmaları iyi bilinmelidir.
- Ölçütleri toplarken ölçütlere bulaşanların farkında olmalıdırlar. Burada çok ünlü iki "H" vardır (Heisenberg etkisi ve Hawthorne etkisi.) Hawthorne etkisi bir araştırma esnasında gözlendiklerinin farkında olan deneklerin doğal davranmayışları olarak açıklanabilir. Heisenberg etkisi ise aynı zamanda gözlenen iki durumdan, birinin ne kadar gözlemlenirse öbürünün o kadar belirsizliğinin artacağını öne süren bir teoridir.
- Ölçütler zararlı olabilir. Daha doğrusu ölçütler yanlış kullanılıyor olabilirler. Buna sürekli dikkat edilmelidir.

Ölçüt belirleme çalışması yaparken 21 adet gerçek proje, COCOMO 81, yapılmış çalışmalar ve 28 tane 3 yıldan fazla proje yönetim tecrübesi olmuş proje yöneticisiyle birebir görüşüldü. Proje yöneticilerinden temel grupları belirlemelerini istedik. Burada açık uçlu olarak sorduk. Bir liste getirip seçmeleri bir sınırlandırma yaratacağı için kullanmadık. Proje yöneticilerinden belirledikleri grupların yanı sıra bu grupların ağırlıklarını da belirtmelerini istedik. Ellerinde bulundan %100'lük bir pastayı bu gruplar arasında paylaştırmalarını istedik. Proje yöneticileri bunları oluşturduktan sonra biz bu grupların isimleri aynı olanları veya anlamları aynı olup farklı isimlendirilenleri aynı gruba dahil edip ortalamalarını aldık. Bu sayede tablo 1'i elde ettik

**Tablo 1.** Ana Grup Sonuçları

| Ana Grup | Önem Ortalaması |
|---|---|
| İşin Büyüklüğü / Ürünün büyüklüğü | 8,15 |
| Ayrılan kaynak | 9,78 |
| Risk | 9,97 |
| Kullanılan teknoloji | 10,12 |
| Yazılım Geliştirme Ortamı | 11,10 |
| Planlar | 7,08 |
| Tahminler | 6,55 |
| Bütçe | 5,78 |
| Ürünün karmaşıklığı | 6,66 |
| Motivasyon | 2,11 |
| Takımdaşlık | 2,54 |
| Tecrübe | 3,31 |

| | |
|---|---|
| Fiziksel Koşullar | 3,44 |
| Çalışanların yetkinlikleri | 4,34 |
| Proje planında değişiklik oranları | 3,13 |
| Proje kapsamında değişiklik oranları | 3,32 |
| Diğer | 2,62 |

Bu tablo sonucunda %95'i oluşturacak 6 temel grupta toparlayabileceğimizi gördük. Buna göre :

- Ürünün karmaşıklığını ürün içinde değerlendirebileceğimize
- Bütçe'yi ürünün içinde değerlendirebileceğimize
- Ayrılan kaynağı ve tecrübeyi ve çalışanın yetkinliklerini kaynak başlığı altında toplayabileceğimizi
- Değişikliklerle ilgili kısımları risk konusunda ele alınabileceğini
- Planlar ve tahminlerin tek başlık altında ele alınabilineceğini gözlemlemiş olduk.

Ana gruplarımızın işin büyüklüğü (ürün), kaynak, risk, teknoloji, ortam ile planlar ve tahminler grupları olabileceğini bu çalışmada gözlemledik.

Bu çalışmadan sonra ana gruplar ve her bir alt grup için proje yöneticilerinin önerileri bir başlangıç matrise eklenmiştir. Elde edilen verilen üzerinde dönüşüm matrisi oluşturuldu. Korelasyonlar belirlenip elenmiştir. Matematiksel, istatistiksel ve gözlemsel bir optimizasyon çalışmasından sonra 6 temel grupta toplanmıştır. Bunlar işin büyüklüğü (ürün), kaynak, risk, teknoloji, ortam ile planlar ve tahminler gruplarıdır.

COCOMO'da belirtilen tüm ölçütler bu önerilen ölçüt setinde kullanılmıştır. Bununla beraber bir hiyerarşik yapı getirilmiştir. Bu çalışmada aynı zamanda daha iyi sonuç alabilmek için bir melez model kullanılmıştır

Birçok ölçütün yanı sıra işlev puanı kullanımı ve baştan yapılan tahminlerinde girdi olarak kullanımı melez bir durum yaratmaktadır ve başarı ihtimalini artırmaktadır. Elde ettiğimiz ölçüt kümesi Tablo 2'de gösterilmiştir. 6 Ana ölçüt grubu ve bunlara bağlı 24 ölçütü bu tablodan görebilirsiniz. Bu 24 tane ölçütün bazılarınında alt ölçütleri vardır. Bunlarla beraber aslında toplam 56 tane girdi ölçütü vardır. Bu alt ölçütleri tablo 2'de gösterip tabloyu karmaşık hale getirmemek için tablodan sonra açıklamaya çalıştık.

**Tablo 2.** Önerilen Ölçüt Seti

| Ana Ölçüt | Ölçüt (Metric) |
|---|---|
| İşin Büyüklüğü (Ürün) | 1. Karmaşıklık |
| | 2.İşlev Puan (Function Point) |
| | 3.Önem |
| | 4. Ayrılan Bütçe |
| | 5. Ürünün beklenen özellikleri |
| Kaynak | 6. Çalışanın yeterlilikleri |
| | 7 Çalışanların Projeye katılım oranları |
| | 8. Çalışan Kişi Sayıları |
| | 9. Donanım Durumu |
| Risk | 10.Bütçe Değişme Riski |
| | 11. Çalışan Riski |
| | 12.Donanım Riski |
| | 13.Ürünün tanımının ve kapsamının değişme riski |
| Teknoloji | 14.Yaz. Geliştirme araçlarının kullanım kolaylığı |
| | 15.Yaz. Geliştirme araçlarının kullanım tecrübesi |
| | 16. Yaz. Geliştirme Araçlarının Kullanımı |
| | 17. Modern Programlama Teknikleri |
| Ortam | 18.Ortamın genel özellikleri |
| | 19.Sahiplenilme (Her bir paydaş türünün projeyi sahiplenmesi) |
| | 20.Baskı |
| | 21.Zaman kullanım durumu |
| | 22.Verimlilik durumu |
| Planlar ve Tahminler | 23. Tahmin |
| | 24. Planlar |

**i. İşin Büyüklüğü**

6 temel gruptan biri olan işin büyüklüğünü bulabilmek için işlev puanı kullanmanın yanısıra ürünün karmaşıklığı, önemi, beklenen özellikleri almak büyüklük temel ölçütünü daha doğru olmasını sağlayacaktır. Karmaşıklığın 3 alt ölçütü olarak Veritabanı büyüklüğü, Ürün karmaşıklığı ve yeniden kullanılabilinirlik olarak belirledik. Bunların üçüde COCOMO'da DATA, CPLX ve RUSE olarak geçmektedir.

İşlev puanı bir yazılım uygulamasının büyüklüğünün ölçütüdür. Bu büyüklük işlevsellik ölçümüdür. Yazılım dilinden, metodolojiden, proje takımının yeterliliğinden bağımsızdır. İşlev puanın nerede zorlandığını bilmek önemlidir. İşlev puan bir uygulama için gereken çabayı ölçmek için mükemmel değildir. Ancak işin büyüklüğünü belirleme anlamında önemlidir. Çoğunlukla gerçek hayatta işin büyüklüğünü anlamak için benzetim kullanılır. İşlev puanı kullanmanın iki avantajı vardır. Bize ürünün büyüklüğünü ve ürünün görünen büyüklüğünü verir. Bunu girdi olarak kulanmak faydalı olacaktır. Dolayısıyla oluşturduğumuz ölçüt setine işlev puanını da bir ölçüt olarak ekledik.

Bir projeye ayrılan bütçe o projenin temel parçalarından biridir. Dolayısıyla projenin büyüklüğü ve önemi ile iligli genel bir fakir verir. Bu açıdan ayrılan bütçe büyüklüğünü bir ölçüt olarak kullanmak faydalı olacaktır. Bununda beraber nihai ürünün önemide direk gereken çabayı etkileyecektir. Bu bize toplam baskıyı vermenin yanısıra o proje atanacak çalışanların yetkinlikleri ile ilgili bir bilgi vermiş olacaktır. Hatta yönetimsel destek ile ilgili önemli bir parametre olması dolaysıyla projenin planlanan zamanda bitmesini etkileyecektir.

Ürünün beklenen özellikleri ölçütünün alt ölçütlerini belirledik. Bunlar esneklik, güvenilirlik, kullanım kolaylığı, bakım yapılabilirlik, güvenlik ve dökümantasyon'dur.

**ii.Kaynak**

Kaynağın büyüklüğü ve niteliği direk olarak sonucu etkilemektedir. Basit olarak bir işin maliyetini bulmak istersek ürünün büyüklüğünü ve dolayısıyla gereken kaynağın büyüklüğünün ne olduğunu bilmeye ihtiyacımız vardır.

Çalışan yetkinliklerini ise gruplara göre alt ölçütlerini oluşturduk. Bunlar proje yöneticisin yetkinliği, yazılım geliştiricilerin yetkinliği, system analistlerin yetkinliği ve test edenlerin yetkinlikleridir.

Çalışan kişi sayılarını gruplara göre alt ölçütlerde kapsamak faydalı olacaktır. Yazılım geliştirici sayısı, analist sayısı, test yapanların sayısı ve genelde bir olmasına rağmen olası durumlar için proje yöneticisi sayısı alt ölçütlerdir.

Donanım durumu ölçütünün de alt ölçütleri vardır. Bunlar zaman kısıtı, depolama kısıtı, platform değişkenliğidir. Bunların üçüde COCOMO'da mevcuttur ve adları TIME, STOR ve PVOL'dur.

**iii.Risk**

Risk etkisi riskin oluşma olasılığı ve bunun maliyetinin çarpımı olarak tanımlanabilir. Dolayısıyla ürün ve kaynak risklerini ve etkilerine ihtiyacımız vardır. Dolayısıyla her birinin alt ölçütlerini belirledik. Her birinin iki alt ölçütü vardır. Bunlar olma olasılığı ve olunca yaratacağı etkidir.

**iv.Teknoloji**

Yazılım geliştirme araçları ve tecrübe oldukça önemlidir ve sonucu oldukça önemli bir biçimde etkiler. Yazılım geliştirme araçlarının kullanım kolaylığı yanında yer alan yazılım geliştirme araçlarındaki tecrübeninde alt ölçütünü belirledik. Bunlar platform tecrübesi, uygulama tecrübesi, yazılım dili ve çoklu ortam geliştirme durumu tecrübesidir. Bunlar COCOMO'da PEXP, AEXP ve LTEX,SITE olarak geçmektedir.

**v. Ortam**

Yazılım geliştirilen kurumun yapısı maliyetleri direk etkilemektedir. Bu kurumdaki ortalama gecikme, ortalama sapma, baskı, sahiplenme ve sorumluluk gibi durumlar sonucu direk etkilemektedir. Ortamın genel özellikleri ölçütünün alt ölçütlerini geciken proje oranı, kurumdaki ortalam gecikme oranı, ortalama gecikme miktarı, gecikmenin standart sapma miktarıdır.

Sahiplenme çok önemli bir ölçütdir. Bunun alt ölçütleri proje yöneticisinin, yönetimin, müşterinin, proje çalışanlarının sahiplenme durumları alt ölçütlerdir.

Baskı miktarı projenin gecikip gecikmeyeceğine ve maliyete etkileyen önemli bir ölçütdir. Bunun 3 alt ölçütü vardır. Bunlar pazar baskısı, müşteri baskısı, yönetim baskısıdır.

Zaman yönetimin alt ölçütleri bir çalışanın ortalama gün içinde kesilme (interrupt) sayısı, bir kesilmenin ortalama süresi, kesilme sonrası önceki duruma geçme ortalama süresi, bu kurumda ortalama yapılan fazla mesai süresidir.

**vi. Planlar ve Tahminler**

Uzman görüşü çoğu yerde en çok kullanılan yazılım maliyeti tahmin etme yöntemidir. Bu yöntem mevcut yöntemler arasında hala en başarılı olandır [18]. Bu görüşüde bir ölçüt olarak almak faydalı olacaktır. Bu uzmanların tahmin sapma oranlarınıda bir ölçüt olarak aldığımız takdirde daha doğru sonuçlar alabileceğiz. Tahmin ölçütünün alt ölçütleri tahmini zaman planı, Tahmini yapankişi veya kişilerin otalama sapma oranıdır. Plan ölçütünün alt ölçütü planlanan zaman, yanılma oranı ve hedeflenen zamanda bitmemesi durumunda katlanılabilinirlik şeklindedir.

## 3. Yapay Sinir Ağı Modeli

### 3.1 Algoritma

Bu çalışmada biz çok katmanlı ileri beslemeli ve geri yayılımlı yapay sinir ağı (MLP) ve Elman yapay sinir ağı modellerini kullandık. Birçok denemeden sonra lineer olmayan eğri uydurmada zaten başarısı sabitlenmiş olan Levenberg-Marquardt [5] eğitim algoritması olarak seçilip uygulanmıştır.

Yapay sinir ağı kullanarak yapılan yazılım maliyet tahminleme çalışmalarında genellikle çok katmanlı ileri beslemeli (MLP) ağlar kullanılmıştır. Ancak dinamik sistemleri modellemede yenilenen yapay sinir ağları (RNN) daha iyi sonuç verebilmektedir. RNN kısmen veya tamamen bağlı olabilir. Elman ağı [5], bilinen en iyi kısmi-bağlı RNN'lerden biridir. Bu çalışmada geri yayılımlı Elman ağ modeli eğim iniş yöntemi kullanarak eğitilmesi de çalışılmıştır.

Her iki modelimizde hem COCOMO '81 hem de YEEM kullanmıştır. Ölçüt veri kümesi ise bu modele girdi olarak verilmiştir.

### 3.2 Veri İşleme

*a. Rasgelelilik ve istisnaların kontrolü*

Henüz herhangi bir işlem yapılmaksızın verinin rasgeleliğini ve istisnaların incelenmesi oldukça önemlidir. Literaturde rasgeleliği ve istisnaları bulmaya yarayan bir çok test yöntemi vardır. Bu testler temel olarak teorik ve deneysel olarak 2 kategoriye ayrılabilinir .Bu çalışmada eğitim ve test veri kümesi rasgelelik kontrolü "Run" testi [10] ile yapılmıştır. Yaptığımız çalışmalarda COCOMO '81'de 3 tane istisnai veri olduğu gözlemlenmiştir. COCOMO '81 63 adet projeden oluşmaktadır. Bu 3 tane istisna verisini çıkarılmasıyla geriye kalan 60 tane veri ile çalışma yapılmıştır. YEEM için ise elimizde 109 adet veri mevcut idi. Bunlardan 9 tanesi istisnai olduğu görüldü ve çıkarıldı.

*b. Verinin ön-işleme ve son-işlemesi*

Bu çalışmada veri [0,1] aralığına çevrilerek kullanılmış ve bu aralığıa getirmek için minumum-maksimum normalizasyon [10] yöntemi kullanılmıştır.

*c. Veri Kümesinin organizasyonu*

Bu çalışmada hem COCOMO '81 hemde YEEM veri kümesi kullanılmıştır. COCOMO '81 ile YEEM'i daha iyi ve gerçekçi karşılaştırabilmek için aynı veri sayısı içinde karşılaştırmaya karar verdik.Bu yüzden YEEM'I 60 adet verilik ve 100 adet verilik olarak iki küme olarak değerlendirmeye aldık. Dolayısıyla toplam 3 veri kümesi kullandık. Her veri kümesini eğitim ve test için iki gruba ayırdık. Bu çalışmada kullanılan veri kümeleri, veri sayısı, eğitim ve test için ayrılan veri sayısı tablo 3'de gösterilmiştir. Üçüncü veri kümesinin başarısı aslında bu çalışmada oryata konan modeli gerçek değerini verecektir.

***Tablo 3.*** *Veri Kümelerinin Organizasyonu*

| | Veri Kümesi 1 | Veri Kümesi 2 | Veri Kümesi 3 |
|---|---|---|---|
| Ölçüt Kümesi | COCOMO | YEEM | YEEM |
| Veri Sayısı (Proje Sayısı) | 60 | 60 | 100 |
| Eğitim kümesi | 45 | 45 | 75 |
| Test kümesi | 15 | 15 | 25 |

## 4.Uygulama Sonuçları

### 2.3 Değerlendirme Kriteri

Bir modeli test etmek için ve ne kadar iyi çalıştığı ölçmek için hata fonksiyonunu tanımlamak gerekiyor. Literatürde bu konuda az olan çalışmalarda genel kabul görmüş değerlendirme kriteri Ortalama bağıl hatadır (MMRE). Ortalama bağıl hata, bağıl hataların toplamının veri kümesinde bulunan toplam sayıya bölünmesi ile bulunabilir (1). Aşağıda gösterilen hesaplamada "N" kullanılan veri kümesinin veri sayısıdır.

$$MMRE = \frac{1}{N}\sum_{i}\frac{\left|Gerçekleşen_i - Hesaplanan_i\right|}{Gerçekleşen_i} \quad (1)$$

MMRE yeteri kadar iyi olmasına rağmen insan algısı için yüzdesel gösterimide faydalı olacaktır. Bu yüzden MMRE'nın 100 katınıda tablolarda göstermekteyiz.

### 2.4 Uygulama Sonuçları

3 veri kümesi için hem MLP hem de Elman ağları kullanmak suretiyle bu çalışma için 6 değerlendirme yapılmıştır. Çalışma sonuçları tablo 4'de gösterilmiştir.

***Tablo 4.*** *COCOMO ve YEEM veri kümeleri ile MLP ve Elman çalışmaları sonuçları*

| Yapay Sinir Ağı Modeli | Veri Kümesi | Veri Sayısı (Proje Sayısı) | MMRE | Ort. % Hata |
|---|---|---|---|---|
| MLP | COCOMO | 60 | 1.805 | 180.50 |
| MLP | YEEM | 60 | 0.280 | 28.00 |
| MLP | YEEM | 100 | 0.207 | 20.70 |
| ELMAN | COCOMO | 60 | 1.312 | 131.20 |
| ELMAN | YEEM | 60 | 0.240 | 24.00 |
| ELMAN | YEEM | 100 | 0.191 | 19.10 |

### 2.5 Çapraz Onaylama (Cross Validation)

Hata hesaplama teknikleri genel performansı göstermede yardımcı olmaktadır [2]. Bu çalışmada v-kat çapraz onaylama (v-fold cross validation) kullanılmıştır. v-kat çapraz onaylama ile elimizdeki veriyi daha iyi kullanmam sağlamaktadır. Örneğin 15-kat çapraz onaylamada, verinin 1/15'ini ayırıp daha sonra modeli aynı prosedüre göre yeniden inşaası yapılmaktadır ve bu işlem 15 kez yapılmaktadır. Bu çalışmada 5,10 ve 15-kat çapraz onaylama yapılmıştır. Tablo 5'te 5,10 ve 15-kat çapraz onaylama çalışmasının sonuçlarını görebilirsiniz.

***Tablo 5.*** *MLP kullanılarak Yapılan Çapraz Onaylama Sonuçları*

| | Veri Kümesi 1 | Veri Kümesi 2 | Veri Kümesi 3 |
|---|---|---|---|
| Veri Kümesi | COCOMO | YEEM | YEEM |
| Veri Sayısı (Proje Sayısı) | 60 | 60 | 100 |
| Ort.% Hata (5-fold cv) | 189.8 | 21.66 | 12.89 |
| Ort.% Hata (10-fold cv) | 189.0 | 22.26 | 12.01 |
| Ort.% Hata (15-fold cv) | 166.4 | 18.97 | 9.57 |

## 5.İlgili Çalışmalar ve Karşılaştırma

Yapay sinir ağı kullarak yazılım maliyet tahminleme konusunda az sayıda çalışma yapılmıştır. Bu bölümde en başarılı olanları el almaya çalışacağız. Wittig and Finnie [11] bu konuda bir çalışma yapmışlardır. Bu çalışmada çok katmanlı perseptron ve veri olarak Desharnias ve Avusturya ölçüt birliği verilerini (ASMA) kullanmışlardır. İki veri kümesinide 3 defa test etmişler ve 10 tane projeyi rasgelelik uygunsuzluğu nedeniyle çıkarmışlardır. MMRE değeri olarak Desharnias için %29 ve ASMA için %17'ye ulaşmışlardır. Bu konuda yapılmış önemli çalışmalar tablo 6'da gösterilmiştir.

***Tablo 6****. Yapılmış çalışmalar*

| Çalışma Adı | Kullanılan Algoritma | Veri Kümesi | Proje Sayısı | Sonuç (MMRE) |
|---|---|---|---|---|
| Venkatachalam [13] | Back-Propagation | COCOMO | 63 | |
| Wittig & Finnie [11] | Back-Propagation | Desharnais/ASMA | 81/136 | %17 |
| Jorgenson [14] | Back-Propagation | Jorgenson | 109 | %100 |
| Serluca [15] | Back-Propagation | Mermaid-2 | 28 | %76 |
| Karunanithi [17] | Cascade-Correlation | | | |
| Samson [12] | Back-Propagation | COCOMO | 63 | %428 |
| Srinivasan | Back-Propagation | Kemerer ve COCOMO | 78 | %70 |
| Hughes [16] | Back-Propagation | Hughes | 33 | %55 |

Bu çalışma'da kullandığımız Elman oldukça başarılı olmuştur. Gözlemliyebildiğimiz kadarıyla en Wittig & Finnie kadar hatta 15-kat çapraz-onaylama ile oldukça iyi bir sonuca ulaşmıştır.

## 6.Sonuçlar

Bu çalışmada temel amaç yazılım maliyet tahminlemede kullanılabilinecek bir ölçüt kümesi ve bir yapay sinir ağı modeli geliştirmekti. Yazılım maliyet tahminleme çalışmalarında ölçüt kümesinin öneminin ihmal edilerek genellikle mevcut kümelerin kullanıldığını ve bu kullanımın sonuca ciddi biçimde etkilediği görülmüştür. Yeni bir çalışma ile yeni bir ölçüt kümesi oluşturulmuştur. Bu ölçüt kümesi oluşturduktan sonra yapay sinir ağı çalışmalarına geçilmiştir.

Yazılım maliyet tahminleme için yapay sinir ağları kullanımı konusunda literatürde bazı çalışmalar vardır. Bu çalışmalarda genellikle COCOMO '81 veri kümesinin kullanıldığını görmekteyiz. Yapılan çalışmaların sonuçları günümüz için hala kabul edilebilinir, kullanılabilinir seviyelere henüz gelmemiştir.

Bu çalışmada karşılaştırma yapabilmek için hem COCOMO '81 hem de yeni oluşturduğumuz YEEM ölçüt kümelerini ayrı ayrı kullandık. COCOMO '81 ile yaptığımız çalışmalarda yapay sinir ağı kullanarak yazılım maliyet tahminlemesi için bu ölçüt kümesinin yetersiz kaldığını gözlemledik. COCOMO '81 ile bir kaç değişik yapay sinir ağı modeli denememize rağmen sonuç olarak kullanılabilir seviyeye gelmedi.

Oluşturduğumuz yeni ölçüt kümesi olan YEEM'i kullandığımızda kullanılabilinir seviyede iyi sonuçlar alınmaktadır. Yeni ölçüt kümesi ve verisi MLP ve Elman yapay sinir ağları kullanılarak test edilmiştir. Elman ağındaki başarı MLP'ye nazaran daha iyi seviyelerdedir. Ancak MLP'de çapraz-onaylama (cross-validation) yaptığımızda hata oranını %15'in altına indiğini görmekteyiz.

Değişiz boyutta ölçüt kümesi ile çalışıldığından veri sayısının modelin başarısı için önemli bir etken olduğuda görülmektedir.

## KAYNAKLAR


[1] Briand L., Morasca S. and Basili V. (2002). An Operational process for goal-driven definition of measures. IEEE Transactions on Software Engineering, 30(2), 120-140.

[2] Fenton N. and Pfleeger S. (1997). Software Metrics: A Rigorous Approach. 2nd. edition. London. Chapman & Hall.

[3] Weyuker E.J. (1988). Evaluating Software Complexity Measures. IEEE Transactions on Software Engineering. 14(9). 1357-1365.

[4] Whitmire S. (1997). Object Oriented Design Measurement. John Wiley & Sons. Inc.

[5] Reed, R. D. and Marks, R. J.: Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks. MIT Press. (1999)

[6] Poels G. and Dedene G. (2000). Distance-based software measurement: necessary and sufficient properties for software measures. Information and Software Technology. 42(1). 35-46.

[7] Krantz D., Luce R.D., Suppes P. and Tversky A. (1971). Foundations of Measurement. Vol. 1. Academic Press. New York.

[8] Leung, H., Fan, Z.: In Handbook of Software Engineering and Knowledge Engineering (Ed,Chang, S. K.). Volume 2 World Scientific. (2002)

[9] Knuth E. D.: The art of computer programming. 2nd ed. Addison-Wesley. (1981)

[10] Pyle, D.: Data Preparation for Data Mining. Morgan Kaufmann. (1999)

[11] Wittig, G., Finnie, G.: Estimating Software Development Effort with Connectionist Models. Information and Software Technology. Volume 39 (1997) 469-476

[12] Samson,B.,Ellison,D., Dugard P.: Software Cost Estimation using an Albus Perceptron(CMAC). In Proc Eight International COCOMO Est. Meeting. Pittsburgh. (1993)
[13] Venkatachalam, A.R.: Software Cost Estimation Using Artificial Neural Networks. International Joint Conference on Neural Networks. Nagoya. (1993)
[14] Jørgensen M. and Shepperd M.: A Systematic Review of Software Development Cost Estimation Studies. IEEE Transactions on Software Engineering. (2006)
[15] Serluca, C.: An Investigation Into Software Effort Estimation using a Back-Propogation Neural Network. M.Sc. Thesis. Bournemouth University. (1995)
[16] Hughes, R.T.: An Evaluation of Machine Learning Techniques for Software Effort Estimation. University of Brighton. (1996)
[17] Karunanithi, N.,Whitley, D.,Malaiya, Y.K.: Using Neural Networks in Reliability Prediction. IEEE Software. Volume 9(4) (1992) 53-59
[18] Heemstra F. J. (1992), "Software cost estimation", Information and Software Technology, vol. 34, no.10.