T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİ MADENCİLİĞİ İLE YAZILIM MÜHENDİSLİĞİ DERSİ PROJELERİNİN İYİLEŞTİRİLMESİ

PINAR CİHAN

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

DANIŞMAN
PROF. DR. OYA KALIPSIZ

İSTANBUL, 2013

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

VERİ MADENCİLİĞİ İLE YAZILIM MÜHENDİSLİĞİ DERSİ PROJELERİNİN İYİLEŞTİRİLMESİ

Pınar CİHAN tarafından hazırlanan tez çalışması 23.07.2013 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Prof. Dr. Oya KALIPSIZ

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Prof. Dr. Oya KALIPSIZ

Yıldız Teknik Üniversitesi _____

Prof. Dr. Selim AKYOKUŞ

Doğuş Üniversitesi _____

Yrd. Doç. Dr. Yunus Emre SELÇUK

Yıldız Teknik Üniversitesi _____

ÖNSÖZ

Yazılım Geliştirme derslerinden olan *Sistem Analizi ve Tasarımı*, *Yazılım Mühendisliği* ve *Çevik Yazılım Geliştirme* derslerinde öğrenciler ders kapsamında projeler geliştirmektedir. Bu projeler gruplar halinde iş paylaşımı yapılarak yürütülmektedir. Dönem sonunda öğrencilere bu projeleri ile ilgili sorulardan oluşan anketler yapılmıştır. Bu anketler hazırlanırken amacımız yazılım mühendisliği ders projelerinin iyileştirilmesi için projelerde öğrencilerin sosyal ve teknik yeteneklerden hangisinde daha çok zorlandıklarını tespit etmek ve sorular arasındaki ilişkiler üzerinden öğrenci profilini gözlemlemektir. Derslerde öğrencilerin zorlandıkları bu alanların iyileştirilmesi halinde hem projelerin kalitesi artacak hem de endüstrinin beklentilerini karşılayan daha verimli yazılım mühendisleri yetişmiş olacaktır. Bu çalışmada, Yazılım Geliştirme derslerinde gerçekleştirilen projeler ile ilgili öğrencilere yapılan anketler değerlendirilerek öğrencilerin yetersiz oldukları alanlar gözlemlenmiştir. Ayrıca sorular veri madenciliği yöntemleriyle sınıflandırılarak en başarılı algoritma tespit edilmiştir. Bunun yanında yine veri madenciliği yöntemlerinden olan birliktelik kuralı kullanılarak sorular arasındaki bağlantılar tespit edilmiştir.

Çalışma süresince bilgi birikimi, görüş ve önerileriyle bana her zaman destek olan değerli hocam Prof. Dr. Oya KALIPSIZ, Yük. Müh. Mustafa Özgür CİNGİZ'e ve çalışmamı bitirmem için moral, motivasyonumu üst düzeyde tutmamı sağlayan eşim Dr. Mehmet Timur CİHAN'a teşekkürü bir borç bilirim.

Temmuz, 2013

Pınar CİHAN

İÇİNDEKİLER

SAYFA

KISALTMA LİSTESİ

ACM	Association for Computing Machinery
AMEISE	A Media Education Initiative for Software Engineering
CMMI	Capability Maturity Model Integration
CPAT	Çatışma Prob Değerlendirme Ekibi
CRDA	Ortak Araştırma ve Geliştirme Anlaşması
ÇYG	Çevik Yazılım Geliştirme
DAÜ	Doğu Akdeniz Üniversitesi
DN	Doğru Negatif
DP	Doğru Pozitif
ER	Entity Relationship (Varlık İlişki)
FAA	Federal Havacılık Yönetimi
IBk	Instance-Based learner
IEEE CS	IEEE Computer Society
MAE	Mean Absolute Error (Ortalama Mutlak Hata)
MLP	Multilayer Perceptron
MSE	Mean Square Error (Hata Kareleri Ortalamasının)
NB	Naive Bayes
RMSE	Root Mean-Squared Error (Kök Hata Kareler Ortalaması)
RaPSEEM	Reuse and Progress Driven Software Engineering Educational Method
SA	Sistem Analizi
SEGV	Yazılım mühendisliği, Grafik ve Görselleştirme
SMO	Sequential Minimal Optimization Algorithm
SVM	Support Vector Machine
SWCEPP	Software Engineering Code of Ethics and Professional Practice
SWEBOK	Software Engineering Body of Knowledge
SWEEP	Software Engineering Education Project
YN	Yanlış Negatif
YM	Yazılım Mühendisliği
YP	Yanlış Pozitif
WEKA	Waikato Environment for Knowledge Analyses

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ÖZET

VERİ MADENCİLİĞİ İLE YAZILIM MÜHENDİSLİĞİ DERSİ PROJELERİNİN İYİLEŞTİRİLMESİ

Pınar CİHAN

Bilgisayar Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Prof. Dr. Oya KALIPSIZ

Yazılımın hayatımızdaki artan önemi ve ülke ekonomisine etkileri, yazılım mühendisliğine olan ilgiyi hem akademik çevrelerde hem de endüstriyel platformlarda artırmıştır. Yazılım uygulamalarının geliştirilmesi son derece pahalı, hataya yatkın ve yoğun çalışma gerektirmektedir. Yazılım projelerinde başarısızlıkların nedeni; planlı bütçenin veya zamanın aşılması, müşteri isteklerinin karşılanamaması, projelerin iptali, düşük güvenilirlik ve kalite sorunları nedeniyle hizmet dışı bırakılmasıdır. Yazılım projelerindeki bu başarısızlıklardan dolayı, endüstri ihtiyaçlarını yeterince karşılayamayan mezunların yetiştirildiğini iddia etmektedir. Bu da akademi ile yazılım endüstrisi arasında boşluk yaratmaktadır.

Tez kapsamında amacımız, yazılım mühendisliği ders projelerinin iyileştirilmesidir. Bunun için, Yıldız Teknik Üniversitesi, *Sistem Analizi ve Tasarımı* dersinden 86 öğrenci, *Yazılım Mühendisliği* dersinden 70 öğrenci ve Namık Kemal Üniversitesi, *Çevik Yazılım Geliştirme* dersinden 33 öğrenci yani toplamda 189 öğrenciye ders projeleri ile ilgili anket yapılmıştır.

Elde edilen bulgulara göre; öğrenciler projelerde dokümantasyon, ekip içi uyum ve süre kısıtlılığı gibi sosyal yeteneklerde yetersizdirler.

Ayrıca veri madenciliğinde sıkça kullanılan yöntemlerden biri olan sınıflandırma yöntemi kullanılarak anket sorularını sınıflandırmada en başarılı algoritma belirlenmiştir. En başarılı sınıflandırma algoritması belirlenirken Doğruluk, Kappa,

Ortalama Mutlak Hata, ve Kök Hata Kareler Ortalaması sonuçları göz önüne alınmıştır. Elde edilen sonuçlara göre; anket sorularını sınıflandırmada en başarılı algoritma C4.5 algoritmasıdır.

Bunun yanında, anket soruları arasındaki beklenmeyen ilişkileri bulmak, gizli bilgileri açığa çıkarmak için, veri madenciliğinde birliktelik kuralı çıkarım algoritmalarından biri olan Apriori algoritması kullanılmıştır.

Anahtar Kelimeler: Yazılım Mühendisliği Eğitimi, Endüstri-Akademi İşbirliği, Eğitimde Veri Madenciliği, Veri Sınıflandırma, Birliktelik Kuralı

ABSTRACT

IMPROVEMENT OF SOFTWARE ENGINEERING STUDIES PROJECTS WİTH DATA MINING

Pınar CİHAN

Department of Computer Engineering

MSc. Thesis

Advisor: Prof. Dr. Oya KALIPSIZ

The effects of the country's economy and the growing importance of software in our lives have increased the interest of software engineering at both of the academic and industrial platforms. The Development of software applications is extremely expensive, error prone and requires hard work. Cause of failures in software projects; the planned budget or deadline exceeded the customer demands can not be met, the cancellation of projects, low confidence and to be out of service due to quality issues. The industry claims that the software engineering graduates are not able to meet their requirements due to the failures of software projects. This also creates a gap between academia and software industry.

Aim of this study to improve software engineering course projects. Therefore we have applied questionnaire totally 189 students about courses projects, 86 students attended from Yıldız Technical University in *System Analysis and Design* course*,* 70 students attended from Yıldız Technical University in *Software Engineering* course and 33 students attended from Namık Kemal University in *Agile Software Development* course.

According to the findings, students are not enough social skills such as documentation, team communication, and project time difficulties in the projects.

Also the most successful algorithm is determined which is one of the methods often used in data mining using the classification for classification survey question. *Accuracy,*

kappa, mean absolute error, root mean squared error results are taken into consideration for determining the most successful classification algorithm. According to the results, J48 algorithm is the most successful algorithm for classification the survey questions.

In addition, find unexpected relationships between the survey questions, to reveal confidential information, apriori algorithm of association rule mining are used, which is one of data mining algorithms.

Keywords: Software Engineering Education, Industry-Academia Cooperation, Educational Data Mining, Data Classification, Association Rule

BÖLÜM 1

GİRİŞ

Günümüzde yazılım mühendisliği yazılım sektöründe en zorlu işlerden biridir ve yazılım mühendislerine olan ihtiyaç hızla artmaktadır. Ancak Yazılım Mühendisliği Endüstrisi, Yazılım Mühendisliği mezunlarının sektör ihtiyaçlarını karşılamadıklarını belirtmektedir. Yeni mezunlar işe başladıkları zaman endüstriler bunların eğitimleri için zaman harcamaktadır. Bu zaman kaybı ve endüstrilerdeki düşük başarı oranlı yazılım projeleri çok ciddi maliyete sebep olmaktadır. Endüstrilerin bu bakışı yazılım mühendisliği endüstrisi ile üniversite arasında boşluk yaratmaktadır. Bu tezde öğrencilere *Sistem Analizi ve Tasarımı (SA)*, *Yazılım Mühendisliği (YM)* ve *Çevik Yazılım Geliştirme (ÇYG)* derslerinde ekip olarak gerçekleştirdikleri projeleriyle ile ilgili anket yapılmıştır. Anketlere Yıldız Teknik Üniversitesi, Bilgisayar mühendisliği Bölümü *Sistem Analizi ve Tasarımı* ile *Yazılım Mühendisliği* dersinde sırasıyla 89 ve 70 öğrenci, Namık Kemal Üniversitesi, Bilgisayar Mühendisliği Bölümü *Çevik Yazılım Geliştirme* dersinden 33 öğrenci yani toplamda 189 öğrenci katılmıştır. Bu anketler 20'şer sorudan oluşmakta olup öğrencilerin teknik ve sosyal yeteneklerini ölçmeye yöneliktir. Bu anketlerdeki teknik yetenekler (nesneye yönelik diyagramlar, yapı diyagramları, varlık ilişki diyagramları, şelale, spiral modeller vb.) ve sosyal yetenekler (ekip iletişimi, dokümantasyon, projelerde zorlanılan alanlar vb.) değerlendirilip incelenmiştir. Bu değerlendirme sonucunda öğrencilerin projelerde hangi alanda yetersiz oldukları tespit edilmiştir. Bu yetersiz alanların tespiti sayesinde yazılım mühendisliği ders projelerinin başarısı artacaktır. Bu da kaliteli projelerin artmasına neden olacağından endüstrinin ihtiyaçlarına daha iyi cevap veren tecrübeli öğrenciler yetişmiş olacaktır. Bu sayede endüstri ile üniversiteler arasındaki bu boşluğun azaltılması hedeflenmektedir.

Veri madenciliği, verilerin elde edilmesinden anlamlı bilginin çıkarımına ve hatta bu bilginin bir eylem planına dönüştürülmesine kadar geçen tüm bir süreci ifade etmek için kullanılmaktadır. Bu süreç, bilginin çeşitli formlarda elde edildiği ve çok sayıda tekniğin kullanıldığı bir süreçtir.

Veri madenciliği, tanımlayıcı nitelikteki faaliyetleri kapsayabileceği gibi öngörü amaçlı modellerin geliştirilmesini de içerebilir. Veri tanımlama ve görselleştirme, kavram tanımlama, bölümleme, sınıflandırma, öngörü ve bağıntı analizi veri madenciliğinin fonksiyonlarını ifade etmektedir.

Günümüzde çok miktarda verinin ve anlamlı bilgi çıkarımı ihtiyacının olduğu eğitimden astronomiye, tıptan meteorolojiye kadar birçok alanda veri madenciliği uygulamalarından yararlanmak mümkündür. Tez kapsamında, anket soruları istatistiksel olarak değerlendirilip öğrencilerin projelerde eksik oldukları alanlar tespit edilmiştir. Veri madenciliğinin çeşitli sınıflandırma algoritmaları incelenerek en başarılı algoritma tespit edilmiştir. Veriler arasındaki beklenmeyen ilişkileri bulmak, gizli bilgileri açığa çıkararak gelecekteki eğilimleri belirlemek için, veri madenciliğinde, birliktelik kuralı çıkarım algoritmalarından biri olan Apriori algoritması kullanılmıştır.

1.1 Literatür Özeti

Literatür taraması yapılırken üniversitelerde bu alanda yapılmış çalışmaları incelemek için *yazılım mühendisliği eğitimi* konusunda yapılmış çalışmalar incelenmiştir. Endüstri, beklentilerini karşılamayan mezunların yetiştirildiği görüşünde olduğundan dolayı yeni *mezunların iş yerlerinde yaşadıkları zorluklar* (yazılım projelerinde yeni başlayan elemanlarının eğitilmeleri) konusunda yapılmış çalışmalar incelenmiştir. Ayrıca üniversiteler ile endüstri arasında yapılan çalışmaların detaylarını öğrenebilmek için *Endüstri akademi işbirliği* ile ilgili yapılmış çalışmalar tez kapsamında incelenmiştir. Bu konularda yapılmış çalışmalar Bölüm 3'te yer almaktadır.

1.2 Tezin Amacı

Bu tez çalışmasında amaç yazılım geliştirme ders projeleriyle ilgili öğrencilere yapılan anketleri değerlendirip, öğrencilerin projelerde eksik oldukları alanları tespit etmektir.

Bu tespitler sayesinde yazılım mühendisliği ders projeleri iyileşerek daha kaliteli ve başarılı olacaktır. Böylelikle öğrenciler mezun olup işe başladıklarında hem yapılan projelerde zorluk yaşamayacaklardır hem de endüstrinin beklentilerini karşılayan daha verimli çalışanlar yetişmiş olacaktır.

Veri madenciliğinin sınıflandırma fonksiyonuna ilişkin tekniklerini gerçek yaşam verileri üzerinde uygulamak, uygulamada elde edilen sonuçları doğruluk, ortalama mutlak hata, kök hata kareler ortalaması ve kappa istatistiklerine göre değerlendirerek hangi tekniğin hangi koşullarda uygulanmasının daha etkin olacağına yönelik önerilerde bulunmaktır.

Ayrıca veriler arasındaki beklenmeyen ilişkileri bulmak, gizli bilgileri açığa çıkararak gelecekteki eğilimleri belirlemek için, veri madenciliğinde, birliktelik kuralı çıkarım algoritmalarından biri olan Apriori algoritması kullanılmıştır.

1.3 Hipotez

Yazılımlarda yapılan hatalar kuruma yüklü bir maliyet getirmekte ve iş kaybına yol açmaktadır. Bu nedenle yazılım mühendisliği eğitimi oldukça önemli olup, öğrenciler mezun olmadan ve önemli tasarım ve uygulama sorumluluklara almadan önce mutlaka bazı pratik deneyimleri kazandırmalıdır. Yazılımlarda yapılan bu hataları azaltmak için yazılım geliştirme derslerinden *Sistem Analizi ve Tasarımı, Yazılım Mühendisliği* ve *Çevik Yazılım Geliştirme* olmak üzere üç ders kapsamında öğrencilere yapılan anketlerin değerlendirilmesi sonucunda, projelerde öğrencilerin sosyal ve teknik yeteneklerden hangisinde daha çok zorlandıklarını tespit etmek hedeflenmiştir. Eksik yeteneklerin tespiti sayesinde öğrencilerin ve eğiticilerin ders kapsamında ve projelerde nasıl bir yol izlemesi gerektiği ortaya çıkacaktır. Bu eksik alanların iyileştirilmesi öğrencilerin başarısını doğrudan etkileyecek, yeni mezunların işe başladıklarında zorlandıkları alanların azalmasına yardımcı olacak ve projelerin kalitesinin artmasıyla birlikte endüstrinin beklentilerini karşılayan daha verimli yazılım mühendisleri yetişmiş olacaktır. Elde edilen veriler bir sonraki yazılım projelerinin analiz ve tasarımda dikkat edilmesi gereken noktaları göstererek süreç adımlarının takibini kolaylaştıracaktır. Ayrıca burada asıl amaç derslerin ve projelerin daha verimli ve kaliteli gerçekleştirilmesidir. Öte yandan akademik tarafındaki sonucuna bakacak olursak

yazılım mühendisliğinde ekip kuruluşunun projelendirilmesi üzerinde onu iyileştiren, proje yönetimi süreç teknikleri konusunda hazırlanmış çalışmalar mevcuttur. Amacımız, bu çalışmalara benzer bir çalışma yaratmaktır.

BÖLÜM 2

YAZILIM MÜHENDİSLİĞİ EĞİTİMİ

Yazılım mühendisliği eğitimi, geleneksel olarak üniversitelerin Bilgisayar Bilimleri ve Bilgisayar Mühendisliği bölümlerinde yer alan dersler ile gerçekleştirilmektedir. Bu programlarda "Yazılım Mühendisliği" adında bir ders bulunmakta, bazı programlarda ise buna ek olarak yazılım mühendisliğinin çeşitli konularını işleyen dersler yer almaktadır. Dünyada birçok üniversitede, Yazılım Mühendisliği Lisans Programları oluşturulmuştur [1]. Yapılan bazı çalışmalar yazılım mühendisliği dersinin eğitim programları ile ilgilidir [2], [3], [4], [5], [6], [7], [8], [9].

Yazılımın hayatımızdaki artan önemi ve ülke ekonomilerine etkileri, yazılım mühendisliğine olan ilgiyi hem akademik çevrelerde hem de endüstriyel platformlarda artırmıştır. Yazılım mühendisliği eğitimi ilgi çekici bir konudur ve bu alanda çok sayıda çalışma yapılmıştır [10], [11]. Günümüzde yazılımlarda karşılaşılan problemler, yazılım mühendisliği eğitiminin önemini vurgulamaktadır. Bu bakımdan, yazılım problemlerinin çözümü ve iyileştirmeler için, gelecekteki yazılım mühendislerinin eğitimine odaklanılması gerektiği, literatürde sıklıkla vurgulanmaktadır [12], [13]. Ayrıca geleceğin yazılım mühendislerinin eğitimi çeşitli boyutlarda incelenmiştir [14], [15], [16], [17], [18], [19], [20].

Günümüzde yazılım geliştirme alanında milyonlarca kişi çalışmakta, yazılım mühendisliğinde yer alan çeşitli konular üzerinde çeşitli konferanslar düzenlenmekte ve yazılım mühendisliği üzerine eğitim programları bulunmaktadır. Ancak yazılım mühendisliği çevrelerinin üzerinde birleştiği bir konu, yazılım mühendisliğinin halen

"genç" bir disiplin olduğu ve yeterli olgunluğa ulaşması için birçok konuda fikir birliğinin sağlanması için zamana gereksinim olduğudur [21].

Yazılım mühendisliğinin olgunlaşma sürecini hızlandırmak için yakın geçmişte, uluslararası meslek kuruluşları olan Association for Computing Machinery (ACM) ve IEEE Computer Society (IEEE CS), "Yazılım Mühendisliği Koordinasyon Komisyonunu" oluşturmuş ve yazılım mühendisliği alanında çeşitli projelere başlamıştır [22]. Bunlar;

- Yazılım mühendisliği çekirdek bilgisinin tanımlanması - Software Engineering Body of Knowledge (SWEBOK),
- Yazılım mühendisliği etiklerinin tanımlanması - Software Engineering Code of Ethics and Professional Practice (SWCEPP),
- SWEBOK ile uyumlu olarak örnek bir eğitim programı tanımlanması Software Engineering Education Project (SWEEP)

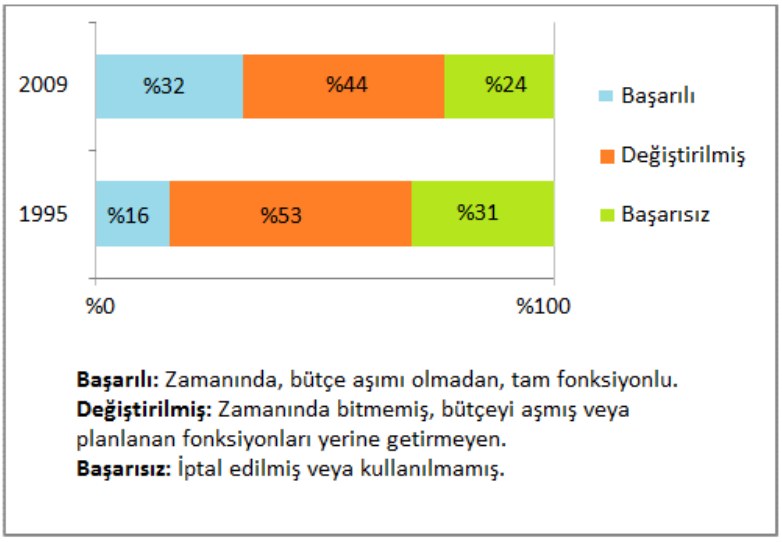
IEEE CS ve ACM, bilgisayar lisans programlarının müfredatlarına yönelik rehberlerin gözden geçirilmesi ve yenilenmesi için ortak bir kurul oluşturmuşlardır. Bilgisayar Müfredatları (Computing Curricula) adı verilen bu etkinlikte, Bilgisayar Bilimleri, Bilgisayar Mühendisliği, Yazılım Mühendisliği ve Bilgi Sistemleri alanlarında çalışmalar yapılmış ve her bir konu için raporlar hazırlanmıştır [23]. Yazılım mühendisliği müfredatıyla ilgili bu grup, ilk olarak bir yazılım mühendisliği lisans programı için aşağıda özetlenen amaçları, bir yazılım mühendisliği lisans programının mezununun sahip olması gereken yetenekler açısından tanımlamıştır:

- Yazılım ürünleri geliştirmek için bir takımın parçası olarak çalışmak,
- Kullanıcı gereksinimlerini belirlemek ve onları yazılım gereksinimlerine çevirmek,
- Çelişen amaçları düzenlemek, maliyet, zaman, bilgi ve organizasyon kısıtlamaları içinde kabul edilebilir uzlaşmalar bulmak,
- Bir veya daha çok uygulama alanı için, etik, sosyal, yasal ve ekonomik ilgileri bütünleştiren mühendislik yaklaşımlarını kullanarak uygun çözümler tasarlamak,

- Yazılım tasarımı, geliştirilmesi, gerçekleştirimi ve doğrulanması için bir temel sağlayan mevcut teorileri, modelleri ve teknikleri anlamak ve uygulayabilmek,
- Tipik bir yazılım geliştirme ortamında etkin olarak çalışmak, gerekli olduğunda liderlik yapabilmek ve kullanıcılarla iyi iletişim kurabilmek,
- Yeni modelleri, teknikleri ve teknolojileri öğrenebilmek.

Kurul, bu amaçlar doğrultusunda, on temel alandan oluşan Yazılım Mühendisliği Eğitimi Bilgi Alanları (Software Engineering Education Knowledge Areas) tanımlamıştır. Yazılım Mühendisliği Eğitimi Bilgi Alanlarında tanımlanan bilgi alanları, bir müfredat oluşturmamakta, sadece yazılım mühendisliği eğitim programlarının tasarlanması ve gerçekleştirilmesi için bir temel sağlamaktadır. Yazılım Mühendisliği Eğitimi Bilgi Alanları şu şekilde açıklanmıştır [1]: *Yazılım Mühendisliğinin Temelleri*, *Profesyonel Uygulama*, *Yazılım Gereksinimler*, *Yazılım Tasarım*, *Yazılım Oluşturma*, *Yazılım Sınama ve Doğrulama*, *Yazılım Gelişimi*, *Yazılım Süreci*, *Yazılım Kalitesi* ve *Yazılım Yönetimi*.

1995 yılında ABD'de yapılan bir çalışmada projelerin ortalama %16'sının başarılı olduğunu, %31'inin tamamlanamadan iptal edildiğini, tamamlanan projelerin ise yarıdan fazlasının tahmin edilen bütçenin %189 üzerinde bir maliyet ile tamamlandığını göstermektedir [24]. 1999 yılında yapılan araştırmada bu projelerin yaklaşık %75'inin ya zamanında tamamlanamadığını ya da iptal edildiğini göstermektedir [25]. 2009 yılında yapılan diğer bir araştırmaya göre ise başarılı projelerin ortalaması %32'ye çıkmasına rağmen halen düşük seviyededir. Başarısız projelerin oranı %24, değiştirilmiş projelerin oranı ise %44'tür [26]. Değiştirilmiş projeler başarısız olarak nitelendirilebilir. Çünkü değiştirilmiş projeler, geç bitirilebilmiş, masrafı planlanandan yüksek veya planlanan işlemleri gerçekleştirmeyen projelerdir. Bu projelerin iyi planlanamaması kuruma yüklü bir maliyet getirmekte ve iş kaybına yol açmaktadır. Şekil 2.1 1995 ve 2009 yılında yapılan proje değerlendirme sonuçlarını göstermektedir.



Şekil 2.1 1995 ve 2009 yılında yapılan proje değerlendirme sonuçları

Bu başarısız projeler endüstriler için ciddi maliyetlere sebep olmaktadır. Yazılım mühendisliği eğitiminin ilerideki sorunları düşünüldüğünde, günümüz eğitim programları hazırlanırken göz önüne alınması gerekenler şöyle özetlenebilir [27]:

- Programların öğrenciler için cazip olacak şekilde hazırlanması,
- En etkili şekilde eğitime odaklanılması,
- Endüstri ile iletişimin daha aktif gerçekleştirilmesi,
- İleriye yönelik öğretim programlarının tanımlanması,
- Eğitimin mevcut öğrencilerin koşullarına göre gerçekleştirilmesi,
- Eğitime daha çok gösterim odaklı bir yapı kazandırılması,
- Yazılım mühendisliği eğitimi için temel altyapı gerektiğinin kabul edilmesi,
- Yazılım mühendisliği eğitim araştırmalarının nitelik ve saygınlığının arttırılması.

Yazılım mühendisliği günümüzde yazılım sektöründe en zorlu işlerden biridir. Ancak Yazılım Mühendisliği Endüstrisi, Yazılım Mühendisliği mezunlarının ihtiyaçlarını karşılamadıklarını belirtmektedir. Bir kısım çalışmalar yazılım mühendislerinin iş hayatına en uygun şekli ile hazırlanmalarını ön plana almıştır [28], [29], [30], [31], [32], [33], [34], [35], [36], [37]. Ayrıca yapılan çalışmaların birçoğunda üniversitelerden mezun olan öğrencilerin deneyimsiz olduklarını, iletişim, işbirliği, kişiler arası uyum,

çözüm üretme ve yönlendirme becerilerinin iyi olmadığı gözlemlenmiştir [38], [39], [40], [41], [42]. Yeni mezunlar işe başladıkları zaman endüstriler bunların eğitimleri için zaman harcamaktadır. Bu zaman kaybı ve sanayilerdeki düşük başarı oranlı yazılım projeleri çok ciddi maliyete sebep olmaktadır. Yazılım mühendisliği eğitimi, yazılım mühendisliği iş gücünün kalitesini doğrudan etkileyen bir fonksiyondur. Yazılım mühendisliği eğitimi söz konusu olduğunda, sanayinin ihtiyaçları ve üniversitenin sundukları arasında bir boşluk vardır. Birçok çalışmada endüstri ile üniversite arasındaki boşluğun giderilmesi için aralarında işbirliği yaparak ortak projeler yürütülmesi üzerinde durmuştur [38], [43], [44]. Bu yaklaşımların merkezinde takım tabanlı projeler vardır. Bu projeler ya sadece endüstride gerçekleştirilen projelerin taklidi olarak gerçekleştirilmekte [45], [46] ya da [38], [43], [47], [48], [49], [50] çalışmalarda olduğu gibi gerçek dünyadan projeler ile gerçekleştirilmektedir.

Türkiye 9. Kalkınma Planı [51], Beckman ve arkadaşlarına göre [52], Üniversite ile Sanayi arasındaki işbirliğinin önemli faydaları aşağıda özetlenmektedir.

Üniversitenin faydaları:

- Öğrencilerin yazları çalışmaları ve staj yapmaları için daha iyi fırsatlar sağlama,
- Pratik düzeyde uygulamalarda kurumsal sorunları kavrama,
- Fakülteler için sürekli eğitim ve araştırma olanakları sağlama,
- Kurum ortaklarından işbirliği yapılan üniversiteye özel fon sağlama,
- Öğrencilerine ve fakültelerine endüstriyel tecrübe alanları açma,
- Eğitim ve araştırma çalışmaları için finansal destek sağlama,
- Mezunlarına iş alanları yaratma,
- Bölgesel ekonomik gelişmeye katkıda bulunma,

olarak sıralanabilinir.

Endüstrinin faydaları:

- Yazılım kuruluşları personelinin eğitim ve öğretim ihtiyaçlarını analiz ederek bu ihtiyaç doğrultusunda üniversite müfredatını geliştirir,
- Üniversite araştırmalarına, yöntemlerine, araçlarına ve teknolojilere erişim sağlayabilir,
- Çalışanlara uygun maliyetli, özelleştirilmiş eğitim ve öğretim imkanı sağlar,
- Teknolojilerinin genişlemesine ve yenilenmesine olanak sağlar,
- Gelecekteki elemanlarını seçebilir,
- Kendi araştırma kapasitesini artırabilir,
- Şirket prestiji ve imajını yükseltebilir,
- Bölge ve ülkeye karşı sosyal sorumluluk görevini yerine getirebilir,

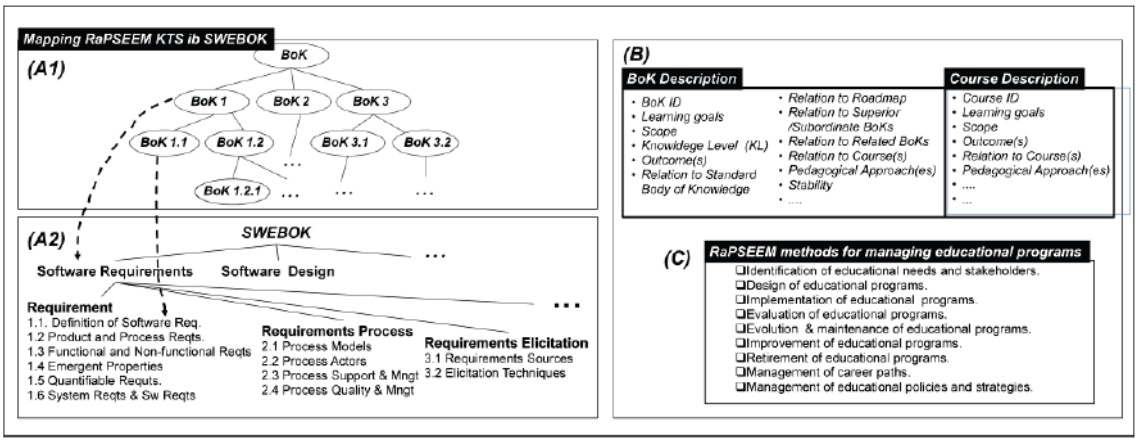
şeklinde sıralanabilinir.

BÖLÜM 3

İLGİLİ ÇALIŞMALAR

3.1 Yazılım Mühendisliği Eğitimi ile İlgili Yapılan Çalışmalar

Kajko-Mattsson [53]: Yazılım mühendisliğinin kapsamı, öğretmek için çok büyük bir alandır. Dolayısıyla, yazılım mühendisliği alanını genişletmek ve doğru eğitim programını seçmek için eğitimcilerin disiplinli ve sistematik bir yönteme ihtiyaçları vardır. Değinilen çalışmada, geliştirilen yöntemin (Reuse and Progress Driven Software Engineering Educational Method (RaPSEEM)) amacı eğitimcilere yardımcı olmak için kendi eğitim programlarını tanımlarken, eğiticileri ve öğrenci becerilerini karşılaştırmak için bir platform oluşturmaktır. Bu yöntemin bir alt kümesinin görüntüsü Şekil 3.1'de gösterilmektedir. Bu yöntem sekiz bileşeni kapsar. Bunlar (1) Yazılım mühendisliği bilgisinin genel organları, (2) Yol haritası, (3) Program tasarım amaçları, (4) Çıktılar, (5) Paydaşlar, (6) Örgüt kültürü, (7) Yazılım mühendisliği yardımcı araçları ve (8) Program tanımı & Yönetim metodolojisi. Bu yöntem teknik ve teknik olmayan sonuçları tanımlar. Teknik yeterliliklerinin özellikleri basit iken, teknik olmayanların özellikleri basit değildir. Bu yöntem teknik olmayan yetkinlikleri böler: (1) *yönetsel yetkinlikler;* analitik, liderlik, planlama, girişimcilik ve idari beceriler içererek öğrencileri çeşitli yönetsel roller üstlenmelerine hazırlamak, (2) *sosyal yetkinlikler;* gözlem formları, iletişim, işbirliği, empati ve saygı içererek öğrencileri her türlü çelişkili duruma hazırlamak [54], ve (3) *deneyim yetkinlikleri*; farklı şartlara ve durumlara yönelik mesleki tutum oluşturmak için öğrencileri hazırlamak.



Şekil 3.1 Geliştirilen yöntemin bir alt kümesinin gösterimi [53]

Bollin vd. [55]: Yazılım mühendisliği eğitimi, bilgi ve yazılım mühendisliği becerilerini kendi meslek hayatı boyunca uygulamak için geniş yelpazede hesaplama gerektirir. Eğitim kurumları ve bireysel mezunlarının gerçek işyerleri arasındaki doğal farklılıklarının yanı sıra müfredat kısıtlamalarından dolayı bir üniversite ortamında derinlemesine tüm konuların kapsanması olanaksızdır. Değinilen çalışmada, bir simülasyon ortamında yazılım proje yönetimi eğitiminin nasıl geliştirebileceğini göstermektedir. Bununla birlikte bu tür bir simülasyonun uygun kullanımında, farklı ders konularının geniş bir yelpazede kapsanması için büyük bir katalizör olmaktadır. Değinilen çalışmada, yaygın olarak kullanılan AMEISE (A Media Education Initiative for Software Engineering) gibi bir simülasyonun yazılım mühendisliği öğretiminde değişen farklı kavramları desteklemek için kullanılabileceğini göstermektedir. AMEISE simülasyonları için standart eğitim ortamı aşağıdaki sıralı aşamadan oluşmaktadır: (1) gerekli birikimin ve teorinin tanıtılacağı veya tekrarlanacağı giriş niteliğinde bir ders, (2) ilk simülasyonu çalıştırmak için planlama, (3) çalıştırılan ilk simülasyonun yürütülmesi, (4) kendi kişisel değerlendirme raporları ve diğer grupların iyi kötü uygulamalarına genel bakış için geri bildirim oturumu, (5) bir öncekinin sonuçlarını iyileştirmek için ikinci simülasyonun çalıştırılması için planlama, (6) çalıştırılan ikinci simülasyonun yürütülmesi, (7) performansları yansıtmak ve yorumlamak için geri bildirim.

Değinilen çalışmadaki derslerde, 200 proje 9 ay içinde 225.000 € bütçe ve son ürünün kalitesi ile ilgili katı şartlar ile yönetilmiştir. Eldeki veriler ile simülasyonun çalıştırılmasıyla elde edilen örnek ekran görüntüleri Şekil 3.2 ve 3.3'te gösterilmiştir. Şekil 3.2(a) "Sistem Özellikleri" ("System Specification") doküman aşamasının gözden

geçirme ve düzeltme olaylarının akışını gösterir. Şekil 3.2(b) özelliklerin ve test aşaması için gerekli kalite ve zamana odaklanmaktadır. Aynı zamanda bu iki aşama için toplam maliyet sağlamaktadır. Şekil 3.2(c) dokümanda algılanmış ve kalan hataların gözden geçirilmesi ve düzeltilmesi için harcanan zamanı gösterir. Şekil 3.2(d) hataların sayısını hatta onların türlerini ve kökeni genişletir.

Group	Author	Reviewer(s)	Corrector	# Errors (1)	(2)	(3)	(4)
sep-03	Richard (+)	Bernd (+), Stefanie (+), Customer (+)	Richard (+)	115	67	61	55
		Bernd (+), Stefanie (+), Customer (+)	Stefanie (-)		26	17	38
		Bernd (+), Stefanie (+)	Christine (-)		0	0	38

Legend: (1) ... Errors contained in the document(2) ... Errors found during review, (3) ... Errors corrected, and (4) Errors remaining in the document

Reviewing skills:Bernd and Stefanie → high, Richard → low, Others → medium
Specifying skills:Richard → excellent, Diana → bad, Others → medium

(a)

Group	Spec (AFP)	Code (AFP)	Spec (PM)	Spec (€)	Tests (PM)	Tests (€)	Total (€)
sep-06	199.33	193.62	1.98	18,697.71	5.12	48,310.90	67,008.60
sep-07	198.99	195.96	1.72	16,616.24	8.04	77,795.42	94,411.66

(b)

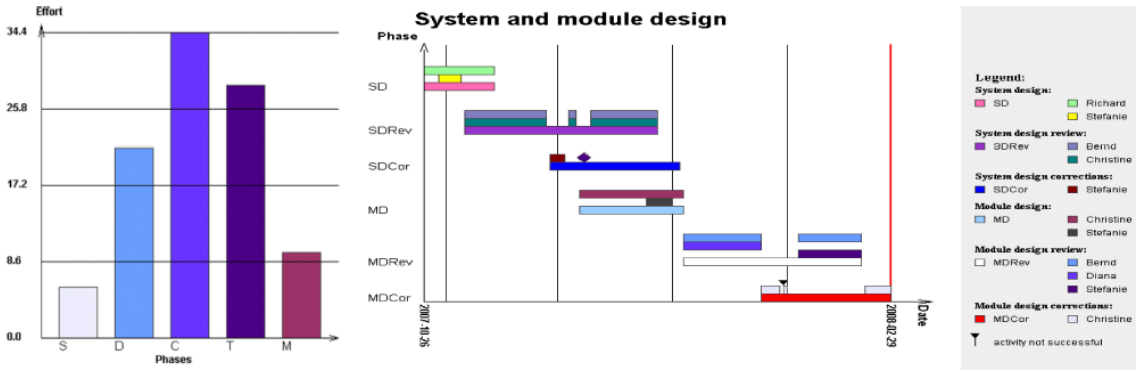
sep-04	Spec	SD	MD	Code	Manual
Length (d)	13.00	18.00	28.00	29.00	18.00
Effort (h)	74.58	56.11	99.84	143.90	28.67
Correction (h)	16.42	39.22	17.64	14.46	16.34
Detected Errors	112.82	57.60	51.80	25.76	55.37
Errors left	17.00	41.00	69.00	57.00	92.00
# Reviewers	3.00	1.00	1.00	1.00	1.00

(c)

sep-04	Spec	SD	MD	Code	Manual	Total
Spec	42	-			-	42
SD	41	17	-	-	-	58
MD	51	16	49	-	-	116
Code	37	9	10	15	-	71
Manual	20	-	-	-	18	38

(d)

Şekil 3.2 Farklı simülasyonların değerlendirme bileşenleri tarafından oluşturulan Tablo ve Diyagram ekran görüntüleri. (a) Diyagram, yorum ve düzeltme sürecinin arkasındaki stratejiyi gösterir. (b) Farklı proje aşamaları için çaba ve gereken kaynakları (zaman, maliyet) gösterir. (c) Tablo, farklı tip dokümanlardaki kodların incelenmesi ve gözden geçirilmesi için harcanan çabayı özetler. (d) Tablo spesifik belgelerde kalan hata türünün ve sayılarını özetler [55]



(a) (b)

Şekil 3.3 Farklı simülasyonların çalıştırılmasıyla geri bildirim bileşeni tarafından elde edilen diyagramların ekran görüntüsü. (a) Diyagram proje aşamasında çaba dağılımını gösterir. (b) Sistem ve modül tasarım aşamasına dahil geliştiricileri gösterir [55]

Albayrak [56]: Çalışmasında İzmir Ekonomi Üniversitesi, Yazılım Mühendisliği bölümündeki öğrencilere bir takım testler yapılmıştır. Bu testlerin amacı üniversitelerde yazılım mühendisi yetiştiren programlara destek olmak için, yazılım geliştirme sürecinin özellikle gereksinim belirleme ve analiz aşamalarına dönük deneyim ve uygulamaları paylaşmaktır. Öğrencilere yapılan test iki kısımdan oluşmaktadır. İlk kısımda zorunlu Yazılım Mühendisliği İlkeleri dersindeki öğrencilere dört bölümden oluşan bir ara sınav yapılmıştır. Çizelge 3.1 Sınav Bölümleri çizelgesi, zorunlu sınavlar kısmında kullanılan ölçme aracı bölümleri hakkında açıklama getirmektedir.

Çizelge 3.1 İzmir ekonomi üniversitesi, yazılım mühendisliği bölümündeki sınav bölümleri [55]

Bölüm	Özellik
1	Kapalı kitap, bireysel yanıt
2	Açık kitap, bireysel yanıt
3	Açık kitap, bireysel yanıt, analiz serbest
4	Açık kitap, grup yanıtı

İkinci kısımda ise öğrencilere sınav verilmemiş, onun yerine yalnızca gönüllü öğrencilere soru sorulmuştur. Çalışmanın bu bölümündeki hedef, yazılım mühendisliği öğrencilerinin analiz yaparken kullanıcı gereksinimlerine ne denli yer verdiğinin izlenmesidir. Bu amaçla, iki ayrı grupta yer alan gönüllü öğrencilerle çalışılmıştır. Birinci grupta yer alan öğrenciler Yazılım Mühendisliği İlkeleri isimli dersi alan öğrenciler, diğer grupta yer alanlar da bu dersi almayan, ancak benzer içerikli dersleri farklı isimlerde alan öğrencilerden oluşmuştur. Bölümlere göre sınav başarıları Çizelge 3.2'de gösterilmektedir.

Çizelge 3.2 İzmir ekonomi üniversitesi, yazılım mühendisliği bölümündeki sınav bölüm başarı ayrıntıları [55]

Bölüm	Özellik
1	%75: Analiz başarılı
2	%88: Aynı soru hatalı çözüm
3	%65: Analizle ilgili kullanıcıya soru sordu
4	%73: Grup çalışmasında daha az başarılı

Sonuçlar incelendiğinde zorunlu sınavlar bölümünde tanımlanan arasınav sonuçları kapalı kitap sınavda doğru yanıt veren öğrencilerin büyük bir bölümünün açık kitap

sınavda kapalı kitaba oranla daha başarısız olduğunu göstermiştir. Diğer çarpıcı bir bulgu grup çalışması ile ilgilidir. Öğrencilerden daha önce sınavın 3. bölümünde tek başlarına çözmeleri istenen sorunun aynısı sınavın 4. bölümünde grup olarak yanıtlanmak üzere sorulmuştur. Öğrencilerin büyük bir bölümünün (%73), grup notları, bireysel olarak aldıkları notlardan göreceli olarak daha düşüktür.

Aybay [45]: Çalışmasında Doğu Akdeniz Üniversitesi (DAÜ) Bilgisayar Mühendisliği Bölümü CMPE-412 Yazılım Mühendisliği dersi, dört yıldır proje ağırlıklı olarak verildiğini belirtmektedir. Dersin toplam dönem notu içinde grup halinde yapılan dönem projesinin ağırlığı %50 olarak belirlenmiştir. Sınavların (bir ara sınav ve bir son sınav) toplam ağırlığı %50 olarak belirlenmiştir. Sınavlarda problem çözme yeteneklerini geliştirmeye yönelik tasarım sorularına en az %50 ağırlıkla yer verilmiştir.

Dönem projesi için uygun olacak konunun seçilmesinde dikkat edilecek iki nokta vardır. İlk olarak, projede öğrencilere mutlaka takım çalışması yaptırılmalıdır. Günümüzde yazılım mühendislerinin karşılaştığı en önemli sorunların başında, proje çalışması için oluşturulan takımlar içindeki uyum ve iletişim sorunu gösterilmektedir. Bunun için projelerin en az 3-4 kişilik öğrenci takımları oluşturularak yapılması önerilmiştir. Bu sayı, yazılım projelerinde gereken kullanıcı arayüzü, veritabanı ve programlama parçaları düşünüldüğünde üçün altında olmamalıdır. Öğrencilerden birinin takım lideri olarak çalışması isteniyorsa, sayı en az dört olmalıdır. İkinci olarak, proje konusu gerçek iş yaşamında o sırada popüler olan konulardan birisi olarak seçilmelidir. Böylece öğrenciler gerçek bir sorunla uğraştıklarını bilecekler, dersin sonunda başarılı bir proje teslim ettiklerinde de yazılım projesi deneyimi yanında önemli bir özgüven kazanacaklardır. Son dört yıl içinde derste verilen proje konuları Çizelge 3.3'te verilmiştir. Çizelge 3.4'de ise on dört haftalık bir dönem için yazılım mühendisliği ders takvimi verilmiştir.

Çizelge 3.3 DAÜ CMPE-412 dersinde son dört yılda verilen proje konuları [45]

YIL	PROJE KONUSU
2002	Lefkoşa Dr. Burhan Nalbantoğlu Hastahanesi için yazılım otomasyonu
2003	Turizm ve tanıtma için kullanılacak kiosk makinelerine program hazırlama
2004	KKTC Milletvekili seçimleri için hazırlık ve seçim sonuçları programı
2005	Lisansüstü programlar öğrencileri için kayıt ve izleme programı

Projeler ve buna bağlı olarak derste işlenen konular için 14 haftalık bir dönem için şöyle bir takvim izlenmektedir:

Çizelge 3.4 On dört haftalık bir dönem için yazılım mühendisliği ders takvimi [45]

Hafta	Proje ile ilgili olarak yapılan işler	Derste işlenen konular
1	Proje konusunun ve dönem takviminin açıklanması	Dersin amaçlarının ve dönem projesinin öneminin anlatılması
2	Proje takımlarının oluşturulması ve sorumlu asistanların belirlenmesi	Yazılım geliştirme süreci, proje yönetiminin temelleri
3	Proje konusu ile ilgili öğretim üyesi ve uzman sunuşları	Yazılım gereksinimleri, proje boyutlarının belirlenmesi
4	Yazılım gereksinimleri belgesinin yazılması	Farklı sistem modelleri, boyut ve maliyet tahmin yöntemleri
5	Proje için ara hedef tarihlerinin belirlenmesi	Yazılım tasarımı, modüler çalışmanın önemi
6	Tasarım -1	İşlevsel çözümleme, kullanıcı ara yüzü tasarımı
7	Tasarım - 2	Arasınav
8	Ara sunuşlar	Yazılım geliştirme yöntemleri, prototip geliştirme
9	Alınan geri besleme ile düzeltme/değişikliklerin yapılması	Yazılım süreç denetimi
10	Yazılım geliştirme - 1	Yazılımın kalitesinin belirlenmesi-ölçülmesi
11	Yazılım geliştirme - 2	Test teknikleri -1
12	Test	Test teknikleri - 2
13	Son sunuşlar ve proragramın çalıştırılması	Tekrar
14	Son sunuşlar ve proragramın çalıştırılması (devam)	-

Verilen proje konularına bakıldığında, çoğu henüz deneyimsiz birer programcı olan öğrencilerin yaklaşık üç aylık bir sürede dört dörtlük profesyonel bir iş ortaya koyamayacağı açıktır. Buna rağmen her yıl sunulan projelerin çoğu oldukça başarılı olmakta, son sunumlarda yapılan denemelerde önceden düşünülmeyen ve test edilmeyen bazı bir kaç aykırı veri dışında genellikle istenenlerin yapıldığı gözlenmektedir.

Teles ve Oliveira [46]: Çalışmasında yazılım mühendisliği lisans dersleri müfredatı değiştirilerek öğrencilerin iletişim ve kişiler arası uyum becerilerini karşılamak için müfredata yeni yollar dâhil edilmesi gerektiğini ifade etmişti. Bu çalışmasında, uzun bir dönemde uç (extreme) programlama dersi ile 20 öğrenciyle proje yapılmıştır. Sonuçlar şu şekildedir; birinci sınıfta öğretmen, öğrencilerin birbirini tanıması için bir dinamik grup önermiştir. Böylelikle sınıf arkadaşlarını tanıdıktan sonra yorum yapıp ve daha iyi sorular sorabilmektedirler. Bu öğrencilerin %63 grup dinamiğini gerçekten yararlı bulmuş ve sınıf tartışmalarının bir parçası olmak onlara daha fazla güven vermiştir. Değerlendirme sistemi kullanılarak öğretmen her hafta öğrencilere başarıları hakkında

geribildirim yapar. Öğrencilerin %83' ü geri bildirimlerin çok yararlı olduğunu ve derse daha fazla motive olarak dersle daha fazla ilgilenmeye başlamışlardır. Bu değerlendirme sisteminin diğer teknik disiplinlere uygulanması gerektiği sorulduğunda %89' u olumlu yanıt vermiştir. Öğrencilerin %94' ü egzersizleri sınıfta yapmanın evde yapmaktan çok daha etkili olduğunu söylemiştir.

Blake [57]: Çalışmasında bir simülasyon ortamında öğrenciler gözlemlenmiş ve deneyimler anlatılmıştır. Georgetown Üniversitesinde yazılım mühendisliği eğitimi Yazılım Mühendisliği I ve Yazılım Mühendisliği II olmak üzere iki ders yaklaşımı olarak geliştirilmiştir. Yazılım Mühendisliği I ders müfredatında yazılım tasarımı üzerinde, yazılım mühendisliği temelleri özellikle nesneye yönelik tasarım vurgulanmaktadır. Yazılım Mühendisliği II dersinde ise yaşam döngüsüne girişin yanında yazılım mimarisi, yazılım tasarım kavramları ile bağlantılı olarak ele alınmıştır. İlk derste projeler bireysel düzeyde gerçekleştirilirken ikinci derste projeler ekip düzeyinde gerçekleştirilmektedir. Yapılan çalışmada yazılım tasarımında yazılım geliştirme ekibine geçildiğinde, öğrenciler tam bir tasarım ürünü geliştirmenin aslında ne kadar faydalı olduğunu görmüşlerdir. Ekipler ile yazılım geliştirme projelerindeki çalışmada kavramsallaştırma, tasarım ve geliştirme ile öğrencilere gerçek dünya tecrübesi verilmektedir. Ayrıca bireysel ve ekip düzeyindeki bu gerçek entegrasyon sayesinde öğrenciler işbirliğinin ve bilgilerin yaşam döngüsü sırasında bir rol veya gruptan diğerine geçişinin önemini anlamışlardır. Bu pratik yaklaşım yazılım mühendisliği lisans öğrencilerine teknik yazılım mühendisliği prensiplerini öğretmek kadar işbirliğine dayalı becerileri de öğretmek için yeni bir yaklaşımdır.

Walker ve Slotterbeck [58]: Çalışmasında bir üniversite içinde yazılım mühendisliği müfredatında daha gerçekçi ekipler kurulmuş ve bunların karşılaştıkları zorlukları ele alarak önerilerde bulunulmuştur. Öğrenciler hafta sonu öğrencileri ve geleneksel öğrenciler diye iki gruba ayrılmıştır. Hafta sonu öğrencilerde endüstri deneyimi ve olgunluk vardır. Geleneksel öğrencilerin ise daha fazla programlama yeteneği vardır. Gruplar 3 veya 4 öğrenciden oluşmaktadır. Öğrenciler projelerinde veritabanı tasarımı, proje yönetimi, arayüz tasarımı yapmışlarıdır. Bu iki grup aynı sınıf çalışma ortamına konulup gözlemlendiğinde mevcut ekipmanları anlama ve kullanımda çeşitlilik, iletişim engelleri ve teknik bilgi dağarcıklarında dengesizlik gözlemlenmiştir. Bu nedenle de

ders dahilinde ekip çalışmasını desteklemeye teşvik için, müfredatta değişiklik yapılarak müfredat boyunca ekip çalışması kültürünü geliştirme hedeflenmiştir.

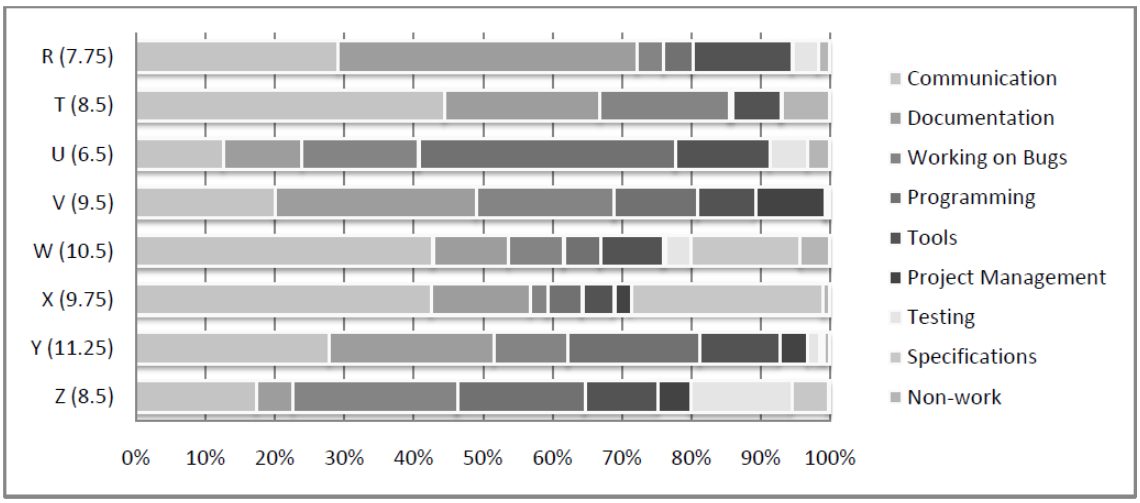
3.2 Yeni Mezunların İş Yerlerinde Yaşadıkları Sosyal Zorluklar ile İlgili Yapılan Çalışmalar

Andrew ve Simon [39], 1 ile 7 ay arasında bir süredir Microsoft tarafından işe başlayan 8 geliştirici seçilmiştir. Bu deneklere programlama, hata üzerinde çalışma, test, proje yönetimi, dokümantasyon ve iletişim gibi bir veya iki görev ve alt görevler atanmaktadır. Kamera ile günlük gözlenen deneklerin verilen görev ve alt görevler ile vakitlerini nasıl geçirdikleri incelenmiştir. Şekil 3.4'de örnek bir gözlem bulunaktadır.

Timestamp	Description	Task Type	Subtask Type
11:45:43 AM	reruns copy script.	Working on Bugs	Reproduction
11:46:18 AM	script done. checks over script output to make sure it looks right. Says that the script is complaining that the files aren't signed. Email with source directory says that they *are* signed. Weird. copied successfully, but binaries aren't signed.	Working on Bugs	Reproduction
11:47:26 AM	Shakes head. Subject T is confused. Team lead says they're signed. But empirical evidence says they're not.	Working on Bugs	Reproduction
11:48:11 AM	Subject T says maybe he wants to sign the binaries himself.		
11:48:36 AM	Subject T mutters to himself "bad bad very bad"		
11:56:23 AM	Subject T goes to A across the hall to ask what's going on.	Communication	Asking Questions
11:56:35 AM	After explaining problem, Colleague A disagrees with Subject T's assessment, comes to Subject T's office and notices that Subject T is copying the wrong architecture binaries to computer. Unsigned binaries are a red herring. Now he copies the right binaries (still said unsigned) and no need to reboot.	Communication	Learning
11:57:27 AM	[Application] now launches just fine.	Working on Bugs	Reproduction
11:57:49 AM	Attempts to repro the bug again. URL works success. repro fails. Subject T expresses confusion why should it repro. Debug binaries and non-debug binaries eliminate repro.	Working on Bugs	Reproduction

Şekil 3.4 T deneğinin görev ve alt görevler bilgileri [39]

Tüm deneklerin görev ve alt görev bilgileri bu şekilde gözlemlenip değerlendirilmiştir. Her deneğin bu görevlerde harcadıkları zaman Şekil 3.5'de gösterilmektedir.



Şekil 3.5 Deneklerin yaptıkları görevler ve bu görevlerde harcadıkları zaman [39]

Elde edilen sonuçlar yeni başlayanların sosyalleşmesi müfredat ve eğitim açısından düşünüldüğünde *ikili programlama* (pair programming), *legitimate peripheral participation* ve *kılavuz* (mentor) bakışı her yerde kabul edilebilirdir. *İkili programlama* ile kodlamayı gözden geçirerek bile bir sosyal unsur oluşacaktır. Bu nedenle akademide ikili programlama yapılması gerektiğini ayrıca yapılan çalışmalar ikili programlamanın tekil başarıyı arttırdığını göstermiştir. *Legitimate peripheral participation* üniversite eğitim deneyimlerinin endüstri pratiklerini karşılamamaktadır bu nedenle akademide endüstri pratikleri uygulanmalıdır. *Kılavuz* yeni gelenlerin sorularına en iyi cevabı verecek ve aralarında sosyal ağ kuracaktır. Sonuç olarak yeni üniversite mezunlarının ilk yazılım geliştirme işlerine başladıklarında temeli, zayıf iletişim becerisi ve sosyal naiflikten kaynaklanan birçok problem yaşadıkları belirtmiştir. Yeni gelenlerin sosyalleşmesi açısından bakıldığında bu durum pek şaşırtıcı değildir. Bu çalışma ile geleneksel müfredat dışında sanayilerin deneyimleri ile daha verimli üniversite öğrencileri hazırlamanın mümkün olabileceği öngörülmüştür.

Begel ve Simon [40], Microsoft şirketinin yazılım geliştirme pozisyonundaki projede farklı ülkelerden 8 öğrenci staj yaparken elde edilen gözlemler sonucunda; tasarım ve geliştirme becerilerinin, iletişimlerinin, işbirliği ve yönlendirme becerilerinin iyi olmadığı gözlemlenmiştir. Endüstrinin ihtiyaçları doğrultusunda yeni bir müfredat hazırlanıp bu müfredatın yazılım mühendisliği eğitiminin hedefi haline getirilmesi önerilmiştir. Tipik bir derste; sınıfta yazılım geliştirme ekibinde 3 veya 5 öğrenci olur. Bu öğrenciler yönetici, geliştirici ve testçi gibi rollere ayrılarak müşteri tarafından

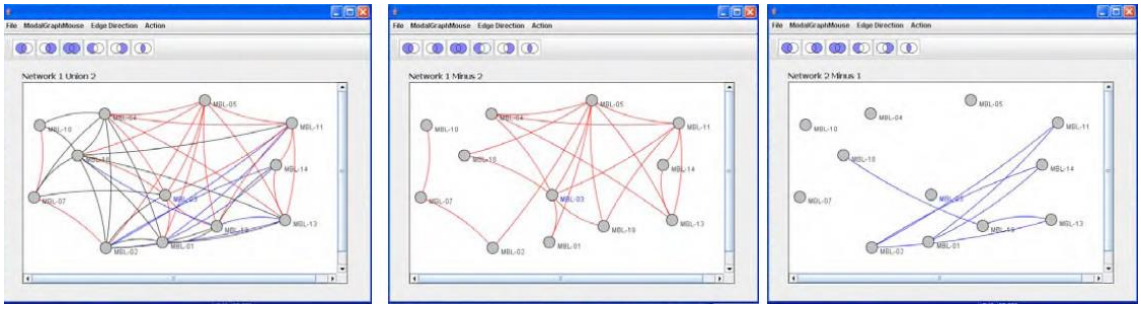
belirlenen ihtiyaçlar doğrultusunda yazılım ürünü geliştirirler. Son ürünün testleri ve dokümantasyonu yapılır. Bu sınıf yeni bir yazılım ürünü simüle etmek, tam tasarım, uygulama ve test döngüsü için tasarlanmıştır. Bunu yaparken de öğrencilere, nispeten yazılımın büyük bir parçası üzerinde birçok kişiden oluşan ekipte nasıl çalışması gerektiğini öğretmektir.

Çalışma sonucunda yeni geliştiricilerin yukarıda tanımlanan sınıftan hayli farklı bir durumda kendilerini bulmaktadırlar. Yeni geliştiricilerin çoğu var olan yazılım geliştirme projelerine en tecrübesiz üye olarak katılırlar ve kolayca erişimi az olan belgeler ile öncesine ait hataları (bugs) ayıklamak için ilk birkaç aylarını harcarlar. Sorunların çoğu programlamada deneyim eksikliği, tasarım veya hata ayıklama değildir. Problemler yeni bir yazılım işinde toplumsal koşullar etrafında merkezlenmiştir.

Yeni bir proje yerine, önceden var olan büyük bir kod temelli projede hataları düzeltme(gerçek veya enjekte edilmiş) ve ek özellikler yazılması öğrenciler için daha değerli bir deneyim olacaktır.

Cleidson R. B. de Souza [59], Global yazılım projelerinde yöneticilerin işini kolaylaştırmak için yazılım bağımlılıkları ve yazılım geliştirme iş koordinasyonu arasındaki ilişkiyi keşfetmek için bir araç geliştirilmiştir. Şekil 3.6 bu araç yazılım geliştirme çalışmalarının doğası hakkında teorik tahminler ve deneysel gözlemlere dayanmaktadır. Bu araç sayesinde yönetici, potansiyel olarak dağıtılan yazılım geliştiricileri izler ve aralarındaki koordinasyon sorunlarını tahmin eder. Bu aracın amacı bağımlılık ve iletişim ağları arasında bir uyumsuzluk olduğunda durumları otomatik tespit etmektir. Yazılım bağımlılıkları ve koordinasyon arasındaki ilişkiyi bulabilmek için, yazılım geliştiricilerin arasında yazılım ve iletişim olaylarının bağımlı olduğu bir parça belirlenmelidir. Bu araç Ariadne [60] kullanarak bağımlı kod parçalarını otomatik bulur, böylece hangi geliştiricinin diğer yazılım geliştiricinin koduna bağlı olduğunun sosyal ağını oluşturmak mümkündür. Mevcut uygulama geliştiriciler arasında alınıp verilen tüm e-postaları alır bir sunucu üzerinden e-posta alışverişlerini kaydeder. Tüm e-postaların toplanması geliştiricilerin iletişim ağını oluşturur. Bu iki sosyal ağ(bağımlılık ve iletişim) bu araç tarafından birleştirilir. Aracın asıl amacı

bağımlılık ve iletişim ağları arasında bir uyumsuzluk olduğunda durumu otomatik tespit etmektir.



Şekil 3.6 Geliştirilen araç tarafından elde edilen üç farklı görüntü [59]

Sillito ve Wynn [61], Bu çalışma yazılım mühendisliğinde sosyal bağımlılık ve zıtlıklar gibi niteliksek gözlemler bildirmektedir. Veriler büyük bir teknoloji şirketindeki yazılım mühendisleri ve yöneticileri ile görüşmelerden, resmi şirket belgelerinden ve toplantılardaki gözlemlerden elde edilmiştir. Çalışma sırasında, üç çeşit süreç iyileştirme çabaları gözlemlenmektedir: Capability Maturity Model Integration (CMMI) temelli çabalar, çevik (agile) veya uç programlama (extreme programming) temelli çabalar ve açık kaynak modelli çabalar. Yazılım mühendislerini anlamak için iş süreçleri ve akışın önemli olduğu düşünülmektedir. Bulgulara göre, modül bağımlılıklarının insanların bağımlılıkları gibi olduğunu ve genellikle eş zamanlı süreç iyileştirme çabaları yazılım mühendislerinin motivasyon çelişkilerini ortaya koymaktadır. Yapılan analizde şirketlerdeki yazılım mühendislerini tamamen anlamak için yazılım sürecinin sosyal yönleri üzerinde durulmaktadır.

Denning [41]: Yeni mezunların deneyimsizliklerinden dolayı genelde organizasyonlar bir yıl süre ile şirket içi eğitim vermek zorunda kalmaktadırlar. Bu nedenle üniversitelerde amacımız öğrencileri bu tür organizasyonlara hazırlamak için çalışılmalıdır. Bunu başarmak için öğrencilere sadece iyi mühendisliği öğretmek değil; nasıl dinlenmesi gerektiğini, işini titizlikle nasıl tamamlayacağını ve kendi kendine nasıl öğrenmesi gerektiğini öğretilmelidir.

3.3 Endüstri Akademi İşbirliği ile İlgili Yapılan Çalışmalar

Almi vd. [42]: Çalışmasının temel amacı iki yönlüdür: Endüstri ve işverenlerin beklentilerini incelemek ve sanayi için yazılım mühendisliği konusunda uzmanlaşmış

son sınıf öğrencilerine yapılan anketleri incelemek. Öğrencilere yapılan anketler iki bölümden ve 11 sorudan oluşmaktadır. Soruların ilk kısmı demografik geçmişleri (Çizelge 3.5, A1-A4) ile ilgiliyken ikinci kısmı mezunların hazır olma durumu (Çizelge 3.5, B1-B7) hakkında bilgi verir. Çalışmanın endüstri tarafında ise üst düzey personeller ile görüşmeler yapılmıştır. Çizelge 3.6'de görüşme sorularının bilgisi bulunmaktadır. Elde edilen sonuçlar ise Çizelge 3.76 ve Çizelge 3.8'de verilmiştir. Analiz sonucunda çoğu mezunun çalışma ortamına hazır olmadıklarını, ilk işe başlamadan önce yönetimsel ve teknik yeteneklerini geliştirmek için ekstra kurslara katıldıklarını göstermektedir. Ayrıca çalışanlara göre bu mezunların yeni fikirler üretmede yaratıcı olmadıklarını gibi sonuçlar da elde edilmiştir.

Çizelge 3.5 Anket soruları bilgisi [42]

Description	Question Number
Characteristic	
Specialisation in study	A1
The age of the respondents	A2
The generalization of the biological studies	A3, A4
The company type that respondents join	B1
Opinion	
Are SE subjects enough to apply in SE related jobs later?	B2
Are you ready to face the real working environment upon graduation?	B3
Did you apply SE subjects and how they were related during industrial training?	B4
Did you learn new things during industrial training?	B5
Does SE specialisation give an advantage in job market?	B6
Expectation	
What do you want to be upon graduation?	B7

Çizelge 3.6 Görüşme soruları bilgisi [42]

Question Number: Description
Opinion
I1: What are the expectations of the employer towards software engineering graduates?
I2: Do most graduates fulfil the expectations of the employer?
I3: What are the aspects that graduates do not have?
Expectation
I4: What are the recommendations to the universities to produce quality software engineering graduates?

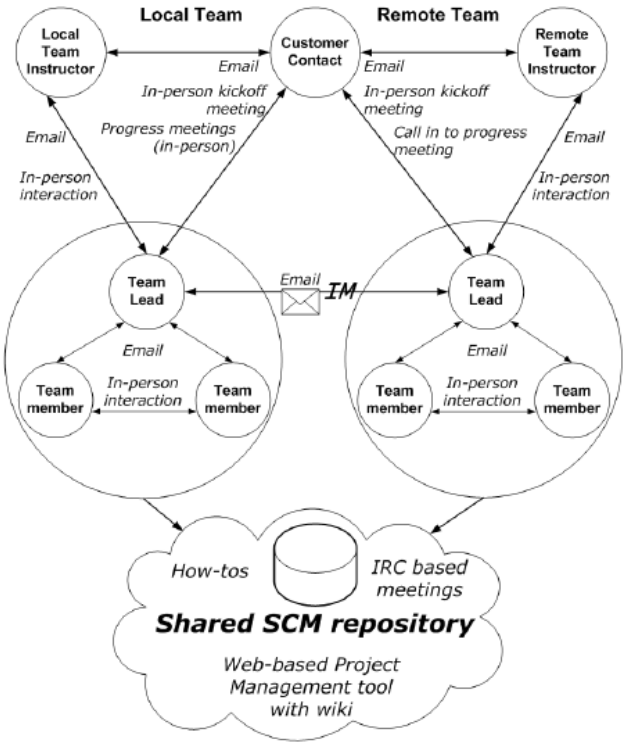
Çizelge 3.7 Mezunların düşünceleri [42]

Question Number: Description	Response Item	F	%
B2: Are SE subjects enough to apply in SE related jobs later?	Yes No	12 8	60 40
B3: Are you ready to face the real working environment upon graduation?	Yes No	12 8	60 40
B5: Did you learn new things during industrial trainings?	Yes No	16 4	80 20
B6: Does SE specialisation give an advantage in the job market?	Yes No	17 3	85 15

Çizelge 3.8 Mezunların yazılım mühendisliği dersi hakkında düşünceleri [42]

Question Number: Description	Response Item	F	%
B4: Did you apply SE subjects and how they were related during industrial trainings?	Yes No	18 2	90 10

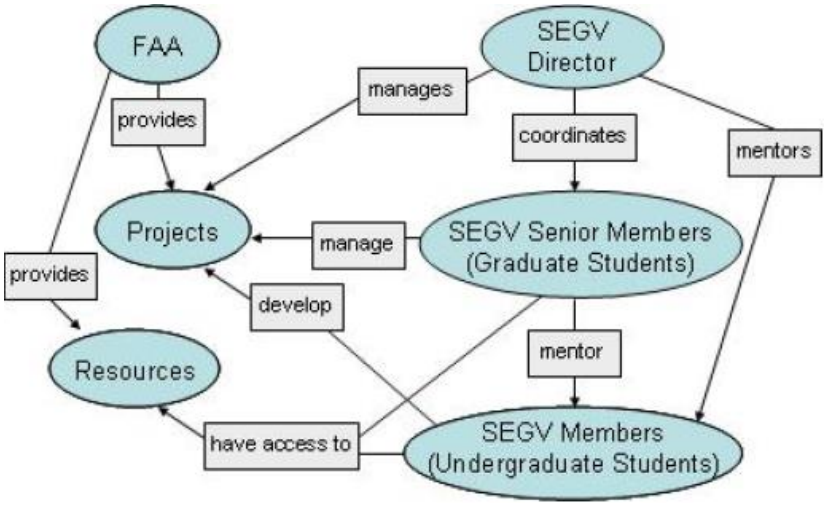
Rusu vd. [43]: Çalışmasında iki üniversite ile işbirliğine girilerek yazılım mühendisliği dersinde gerçek proje üzerinde, gerçek müşteriler ile çalışılmıştır. Her üniversitede seçilen bir proje üzerinde çalışmak üzere yerel ekip kurulur. Yerel ekipler oluşturulduktan sonra yerel ekip, aynı projede seçilen diğer üniversitede eşleştiği ekip veya uzaktan ekip ile birlikte çalışır. Daha sonra yerel ve uzak ekipler arasındaki iletişim için takım lideri belirlenir. Bu organizasyon Şekil 3.7'de gösterilmiştir.



Şekil 3.7 Örnek yerel-uzak ekip etkileşimi [43]

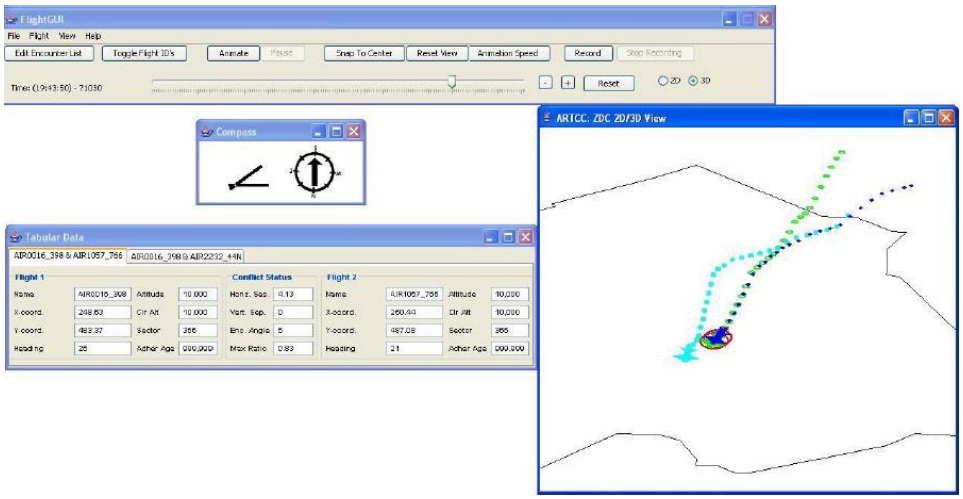
Bu ders ve proje; gereksinim analizi ve spesifikasyonu (1. Evre), Dizayn (2. Evre), Uygulama (3. Evre) ve Test (4. Evre) gibi yaşam döngüsünün tüm önemli evrelerini kapsamaktadır. Her üniversite kendi seçtiği özgün proje üzerinde çalışmıştır. Sunulan bu örnek çalışma ile başarılı bir öğretim tecrübesi ve katılımcı öğrencilerin bakış açısı sonuçları verilmiştir.

Rusu vd. [50]: Çalışmasında Federal Havacılık Yönetimi(FAA)'nin Çatışma Prob Değerlendirme Ekibi (CPAT) ile Rowan üniversitesinden Yazılım mühendisliği, Grafik ve Görselleştirme (SEGV) araştırma grubu birlikte çalışmaktadır. FAA imkânlarını, ekipmanlarını, hizmetlerini, fikri mülkiyet ve personel kaynaklarını Rowan üniversitesi ile paylaşarak FAA ve Rowan üniversitesi arasındaki işbirliği Ortak Araştırma ve Geliştirme Anlaşması (CRDA) ile başlamıştır. Şekil 3.8'de Üniversite ile şirketin arasındaki genel ortaklık yapısı gösterilmektedir.



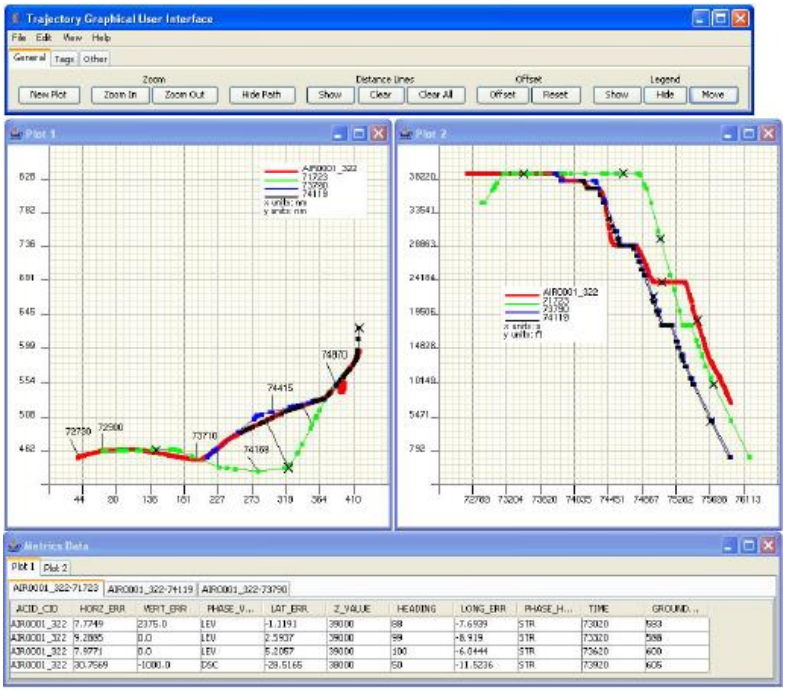
Şekil 3.8 FAA ile SEGV arasındaki genel ortaklık yapısı [50]

Yapılan proje üç yazılım ürününden oluşmaktadır. Her üç yazılım ürününde de ölçeklenebilir nesne tabanlı tasarım (object-oriented designs) vardır. Şekil 3.9'da görüntüsü bulunan Uçuş *Grafiksel Kullanıcı Arayüzü* toplamda 8 lisans öğrencisi tarafından gerçekleştirilmiştir.



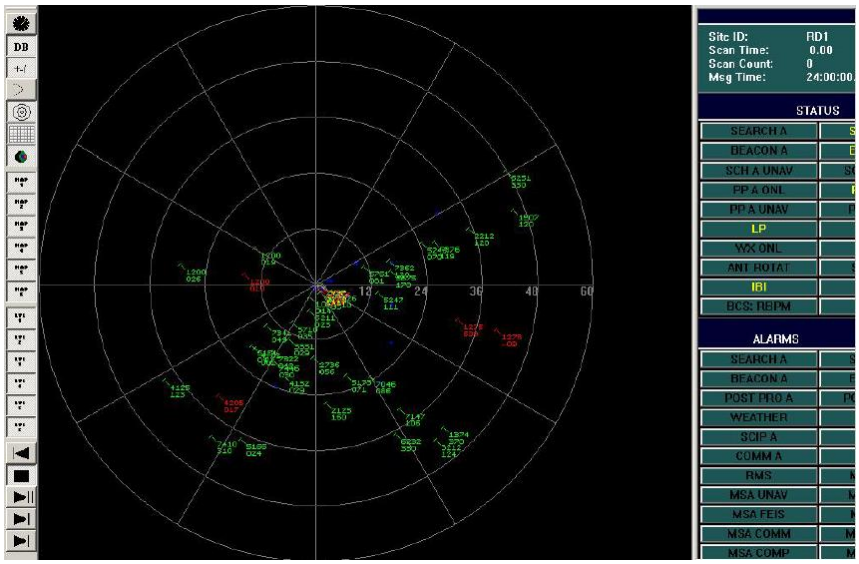
Şekil 3.9 Uçuş grafiksel kullanıcı arayüzü görüntüsü [50]

Şekil 3.10'da görüntüsü bulunan *Yörünge Grafiksel Kullanıcı Arayüzü* toplamda 9 lisans öğrencisi tarafından gerçekleştirilmiştir.



Şekil 3.10 Yörünge grafiksel kullanıcı arayüzü görüntüsü [50]

Ve Şekil 3.11'de görüntüsü bulunan *Radar Simülatörü* ile toplamda 3 öğrenci tarafından gerçekleştirilmiştir.



Şekil 3.11 Radar simülatörü görüntüsü [50]

Gerçekleştirilen bu işbirliği sayesinde öğrenciler gerçek bir proje üzerinde çalışarak deneyim kazanmışlardır. Ayrıca ortaklaşa yapılan bu proje sayesinde, projede yer alan öğrencilerin çoğu ya FAA şirketinin kendisinden ya da FAA'nın firmalarından iş teklifi almışlardır.

Shami [38]: Çalışmasında dünyadaki en büyük bilgi teknolojisi şirketlerinden birinin yazılım geliştirme laboratuvarında gerçekleştirilerek Endüstri-Akademi işbirliğinin çok farklı kültürler, hedef ve teknolojilerde işbirlikçi ortakları bir araya getirmiştir. Bu çalışmanın amacı üniversite çalışanlarını, öğrencileri ve laboratuvar çalışanlarını bir araya getirerek birlikte proje üzerinde çalışmaktır. Çoğu öğrenci yaz tatilinin en az 3 ayını laboratuvarda çalışanlarla birlikte geçirmiştir. Bu süre zarfında üniversite hocası fırsat buldukça laboratuvara uğramıştır. Dersler başladığında ise öğrenciler üniversiteye geri dönmüştür fakat proje üzerinde çalışmaya hocaları eşliğinde devam etmişlerdir. Çalışmaların analiz edilmesi ve işbirliğinin gözlemlenmesi 4 proje grubu üzerinden yapılmıştır ve her grup 6-9 kişiden oluşmaktadır. Tüm gruplar yazılımın farklı alt sistemlerinde çalışmaktır ve her grup kendi sürecinden sorumludur. Bu çalışmanın verileri yazın, 4 ayın üzerinde bir zamanda toplanmıştır. Bu sürede çoğu proje üyesi öğrenci fiziksel olarak laboratuvarda bulunmuşlardır. Öğrencilerin masaları açık-ofis ortamında bulunmakta olup iletişim ve etkileşimin kolay olduğu bir yerdir. Ayrıca bu alan öğrencilerin işbirliği alanı olarak bilinir. Gözlemler sonucunda; üyeler arasındaki teknoloji kullanımı ve sosyal normallikler ilişkisi karmaşıktır. İşbirliği hem zayıf hem

güçlü gruplar belirlendikten sonra zayıf olanlar iyileştirilebilmek için seçilmiştir. Araştırma modeli çalışanların birkaç yıl laboratuvarda bulunmalarını ve üniversitenin bir topluluk oluşturarak bunun karşılıklı olarak hem laboratuvar hem de kendileriyle ilgilenmeleri gerekmektedir.

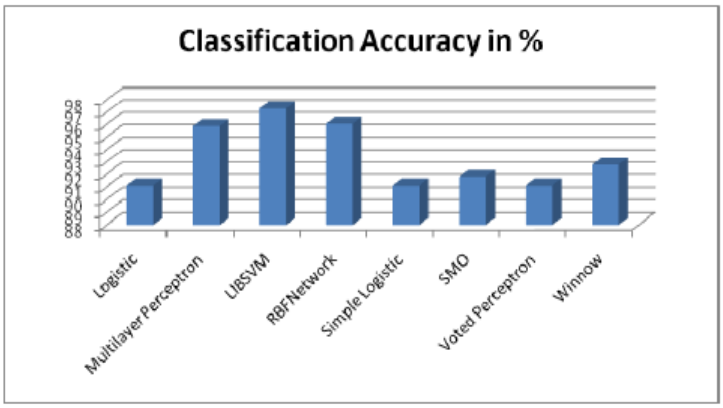
Beckman vd. [44]: Yazılım mühendisliği eğitimi, yazılım mühendisliği iş gücünün kalitesini doğrudan etkileyen bir fonksiyondur. Yazılım mühendisliği eğitimi söz konusu olduğunda, sanayinin ihtiyaçları ve üniversitenin sundukları arasında bir boşluk vardır. Çalışmasında bu boşluğu kapatmak için yazılım mühendisliği programları ve sanayi arasında kapsamlı bir işbirliği önerilmektedir.

3.4 Eğitim Alanında Gerçekleştirilen Veri Madenciliği Uygulamaları

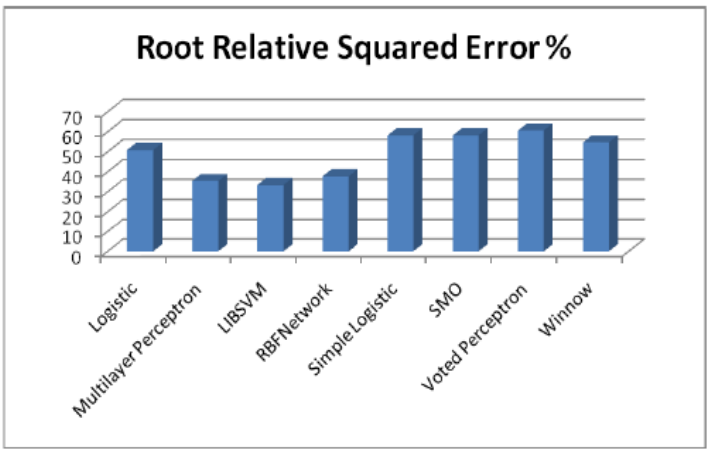
Bhullar vd. [62]: Çalışmasında yükseköğretim kurumlarının karşı karşıya oldukları en büyük zorluklardan biri olan öğrenciye yol gösterme konusunda çalışma yapılmıştır. Kurumlar, hangi öğrencinin hangi derse kaydolmak istediğini ve hangi öğrencinin belirli bir konuda daha fazla yardıma ihtiyacı olduğunu bilmek istiyor. Ayrıca bazen yönetimler öğrenciler hakkında veya yeni sunulan derslerin başarısı gibi daha fazla bilgi öğrenmek istediklerinde tüm bu sorunların cevabı Veri Madenciliğidir. Veri madenciliği daha doğru karar almak için kuruma yardımcı olur ve öğrencinin sonucunu öğrenmek için iyi bir araçtır. Bu çalışmada, weka veri madenciliği aracı ile veri sınıflandırılmıştır. Öğrencilerin sonuçlarını tahmin etmek için J48 algoritması kullanılmıştır. Karar ağacından zayıf öğrenciler ve en fazla kimin başarısız olma şansı var kolayca tespit edilmiştir. Zayıf öğrenciler belirlendikten sonra öğrencinin performansını arttırmak ve başarısızlık sonucunu en aza indirmek için öğrenciler üzerinde çok çalışılabilir.

Agarwal vd. [63]: Çalışmasında bir devlet üniversitesinden alınan öğrenci verileri alınarak çeşitli sınıflandırma yaklaşımları ve karşılaştırmalı bir analiz yapılmıştır. En iyi sınıflandırıcı seçilirken maksimum doğruluk ve minimum kök hata kareler ortalamasına göre seçilmiştir. Çalışmasındaki amaç mevcut eğitim ve stratejik bir yönetim aracı olarak kabul edilebilir iş sistemi için veri madenciliği teknikleri hakkında bir inanç geliştirmektir. Çalışma sonucunda LIBSVM ile radyal bazında çekirdek (radial basis kernel) veri sınıflandırma için en iyi seçenek olarak alınmıştır. Şekil 3.12 sınıflandırma

metotlarının doğruluk yüzdelerini göstermekte, Şekil 3.13 ise bu metotların kök hata kareler ortalamasını göstermektedir.



Şekil 3.12 Weka sınıflandırıcıları için sınıflandırma doğrulukları [63]

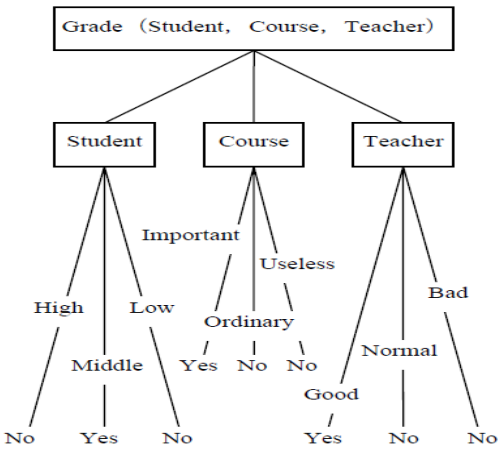


Şekil 3.13 Weka sınıflandırıcıları için kök hata kareler ortalaması [63]

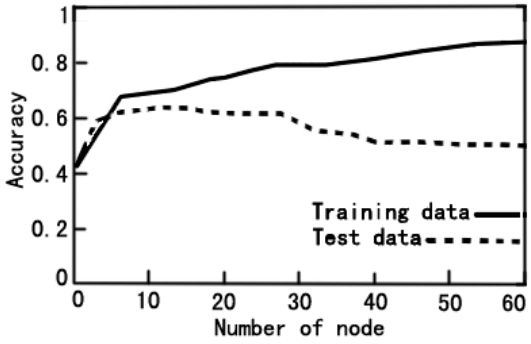
Bunkar vd. [64]: Günümüzde gizli bilgileri ortaya çıkarmak için öğrenciler büyük bir veri setine sahiptir. Veri madenciliği bu gizli bilgileri ortaya çıkarmada yardımcı olur. Makalede, bir kuruma yardımcı olmak için bu veriler üzerinde veri madenciliği tekniklerinden bayes sınıflandırıcı kullanılmıştır. Bu çalışmasında, öğrencilerin notlarına göre hangi sınıflara ayrılması gerektiğini geliştirmek için öğrencilere ve öğretmenlere yardımcı olacaktır.

Sun [65]: Öğrenci öğrenme sonucunu değerlenme sistemi, öğrenci kalitesini kontrol etmek ve izlemek için önemli bir yaklaşım ve araçtır. Bu makale veri madenciliği kullanılarak öğrencilerin öğrenme sonuçlarına ilişkin çıkarımlarda bulunmaktır. Çalışmadaki hedef öğrenci öğrenme sonuç değerlendirmeleri ve öğrenme becerilerini

geliştirmeye yani sonuç olarak daha iyi pratik öğrenmeye hizmet etmek için uygun bir kural keşif yaklaşımı ortaya koymaktır. Şekil 3.14 öğrenci öğrenme sisteminde karar ağacı kullanımını göstermektedir. Şekil 3.15 ise bu karar ağacının eğitimindeki doğruluk oranını göstermektedir.



Şekil 3.14 Öğrenci öğrenme sisteminde karar ağacı kullanımı [65]



Şekil 3.15 Karar ağacı eğitiminin doğruluğu [65]

Bozkır vd. [66]: Çalışmasında ÖSYM tarafından 2008 ÖSS adayları için resmi internet sitesi üzerinden yapılan anket verileri üzerinde veri madenciliği yöntemleri kullanılarak, öğrencilerin başarılarını etkileyen faktörler araştırılmıştır. Bu araştırmada, veri madenciliği yöntemlerinden karar ağaçları ve kümeleme kullanılmıştır. Sonuç olarak okullarda sunulan teknik imkânların ÖSS başarısı üzerinde önemli etkileri tespit edilmiştir. Ayrıca kümeleme analizi sonucunda öğrencinin sahip olduğu sosyal, kültürel ve ekonomik imkânların ÖSS başarısına büyük katkısı olduğu tespit edilmiştir.

BÖLÜM 4

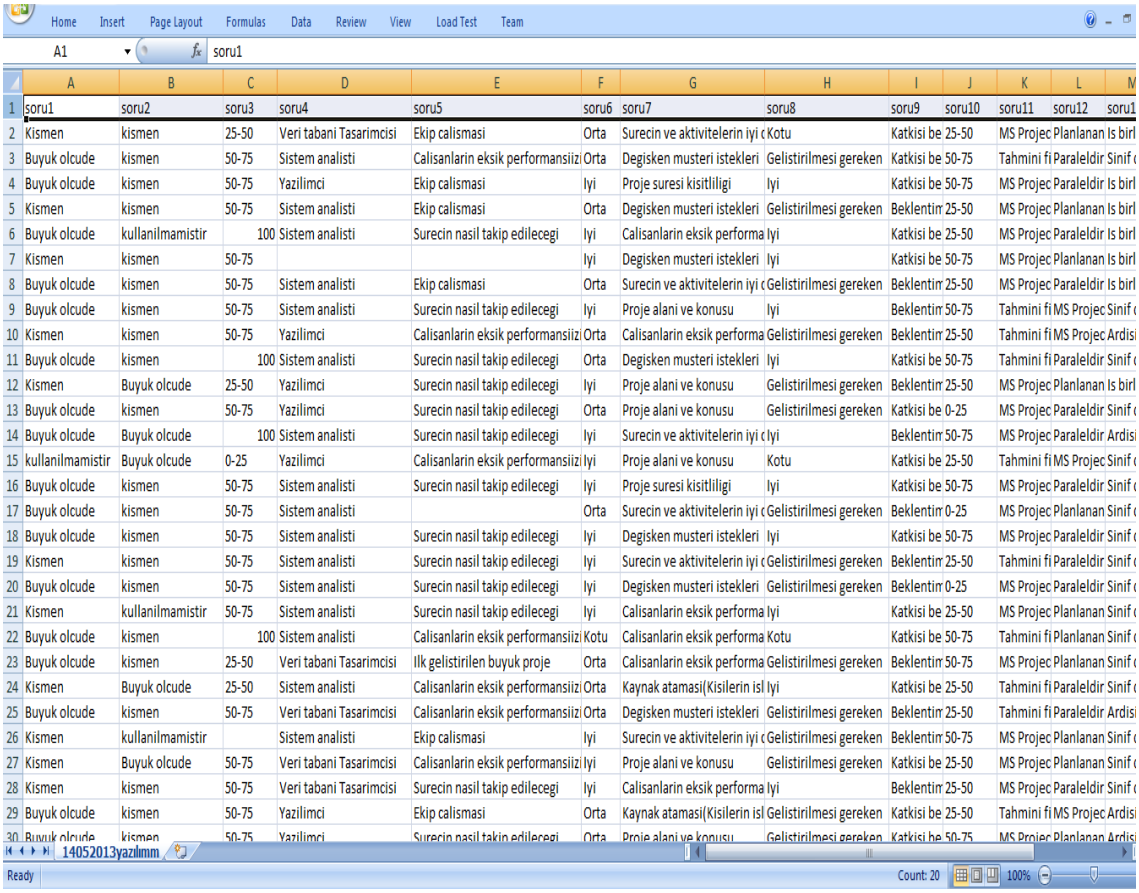
YÖNTEM VE MATERYAL

Veri madenciliği konusunda çok sayıda yöntem ve algoritma geliştirilmiştir. Veri madenciliği modelleri temel olarak sınıflandırma, kümeleme ve birliktelik kuralları şeklinde sınıflandırılabilir. Bu çalışmada veri madenciliğinde sıkça kullanılan yöntemlerden biri olan sınıflandırma ve birliktelik kuralı yöntemleri kullanılacaktır. Verilerin sınıflandırılması için belirli bir süreç izlenir. Öncelikle var olan veritabanının bir kısmı eğitim amacıyla kullanılarak sınıflandırma kurallarının oluşturulması sağlanır. Daha sonra bu kurallar yardımıyla yeni bir durum ortaya çıktığında nasıl karar verileceği belirlenir.

Sınıflandırma için Waikato Üniversitesinde java programlama diliyle geliştirilmiş olan Weka (Waikato Environment for Knowledge Analysis) programı kullanılmıştır. Weka, kullanımı ücretsiz, açık kaynak kodlu, içerisinde pek çok sınıflandırma, regresyon, demetleme, bağıntı kuralları, yapay sinir ağları algoritmaları ve önişleme metotları barındıran, yaygın kullanımlı bir veri madenciliği aracıdır. Tez kapsamında yapılan sınıflandırmalarda algoritmaların başarıları değerlendirilirken doğruluk değeri, ortalama mutlak hata, kök hata kareler ortalaması ve kappa istatistiği kıstasları göz önüne alınmıştır. Veriler arasındaki beklenmeyen ilişkileri bulmak, gizli bilgileri açığa çıkararak gelecekteki eğilimleri belirlemek için, veri madenciliğinde, birliktelik kuralı çıkarım algoritmalarından biri olan Apriori algoritması kullanılmıştır. Ayrıca anket soruları istatiksel olarak incelenerek yazılım projelerinde öğrencilerin yetersiz oldukları alanlar tespit edilmiştir.

4.1 Veri Önişleme

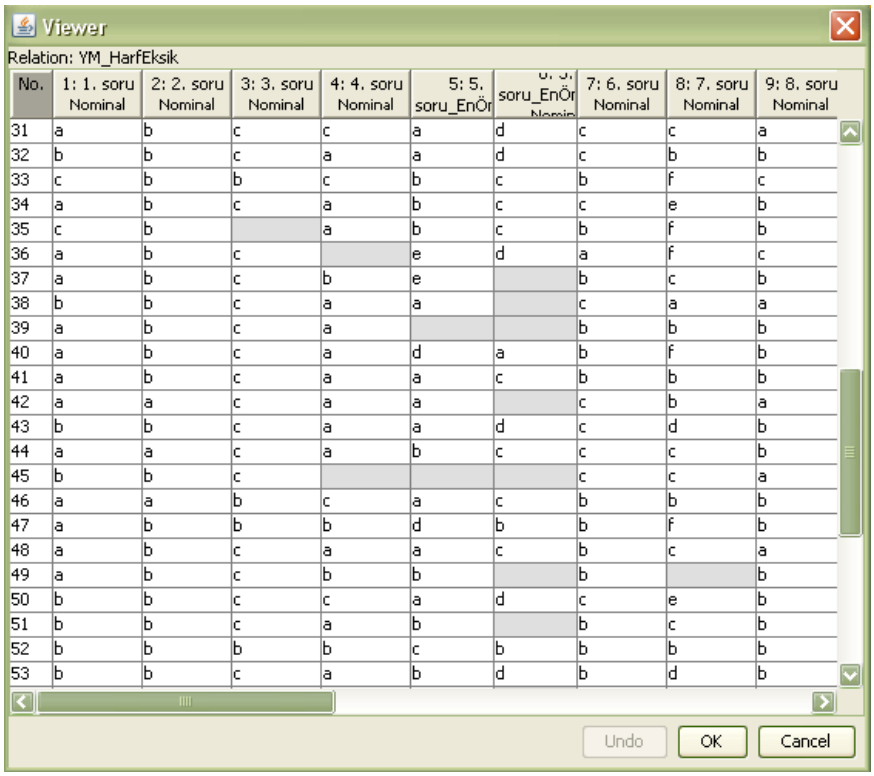
Değerlendirmelerin yapılabilmesi için anket kağıtlarından elde edilen veri seti, dijital ortama aktarılmıştır. Dijital ortama aktarılan anketlerin içeriği Şekil 4.1'de gösterilmiştir.



	soru1	soru2	soru3	soru4	soru5	soru6	soru7	soru8	soru9	soru10	soru11	soru12	soru1
2	Kismen	kismen	25-50	Veri tabani Tasarimcisi	Ekip calismasi	Orta	Surecin ve aktivitelerin iyi c	Kotu	Katkisi be	25-50	MS Projec	Planlanan	Is birl
3	Buyuk olcude	kismen	50-75	Sistem analisti	Calisanlarin eksik performansiiz	Orta	Degisken musteri istekleri	Gelistirilmesi gereken	Katkisi be	50-75	Tahmini fi	Paraleldir	Sinif
4	Buyuk olcude	kismen	50-75	Yazilimci	Ekip calismasi	Iyi	Proje suresi kisitliligi	Iyi	Katkisi be	50-75	MS Projec	Paraleldir	Is birl
5	Kismen	kismen	50-75	Sistem analisti	Ekip calismasi	Orta	Degisken musteri istekleri	Gelistirilmesi gereken	Beklentim	25-50	MS Projec	Planlanan	Is birl
6	Buyuk olcude	kullanilmamistir	100	Sistem analisti	Surecin nasil takip edilecegi	Iyi	Calisanlarin eksik performa	Iyi	Katkisi be	25-50	MS Projec	Paraleldir	Is birl
7	Kismen	kismen	50-75			Iyi	Degisken musteri istekleri	Iyi	Katkisi be	50-75	MS Projec	Planlanan	Is birl
8	Buyuk olcude	kismen	50-75	Sistem analisti	Ekip calismasi	Orta	Surecin ve aktivitelerin iyi c	Gelistirilmesi gereken	Beklentim	25-50	MS Projec	Paraleldir	Is birl
9	Buyuk olcude	kismen	50-75	Sistem analisti	Surecin nasil takip edilecegi	Iyi	Proje alani ve konusu	Iyi	Beklentim	50-75	Tahmini fi	MS Projec	Sinif
10	Kismen	kismen	50-75	Yazilimci	Calisanlarin eksik performansiiz	Orta	Calisanlarin eksik performa	Gelistirilmesi gereken	Beklentim	25-50	Tahmini fi	MS Projec	Ardisi
11	Buyuk olcude	kismen	100	Sistem analisti	Surecin nasil takip edilecegi	Orta	Degisken musteri istekleri	Iyi	Katkisi be	50-75	Tahmini fi	Paraleldir	Sinif
12	Kismen	Buyuk olcude	25-50	Yazilimci	Surecin nasil takip edilecegi	Iyi	Proje alani ve konusu	Gelistirilmesi gereken	Beklentim	25-50	MS Projec	Planlanan	Is birl
13	Buyuk olcude	kismen	50-75	Yazilimci	Surecin nasil takip edilecegi	Orta	Proje alani ve konusu	Gelistirilmesi gereken	Katkisi be	0-25	MS Projec	Paraleldir	Sinif
14	Buyuk olcude	Buyuk olcude	100	Sistem analisti	Surecin nasil takip edilecegi	Iyi	Surecin ve aktivitelerin iyi c	Iyi	Beklentim	50-75	MS Projec	Paraleldir	Ardisi
15	kullanilmamistir	Buyuk olcude	0-25	Yazilimci	Calisanlarin eksik performansiiz	Iyi	Proje alani ve konusu	Kotu	Katkisi be	25-50	Tahmini fi	MS Projec	Sinif
16	Buyuk olcude	kismen	50-75	Sistem analisti	Surecin nasil takip edilecegi	Iyi	Proje suresi kisitliligi	Iyi	Katkisi be	50-75	MS Projec	Paraleldir	Sinif
17	Buyuk olcude	kismen	50-75	Sistem analisti		Orta	Surecin ve aktivitelerin iyi c	Gelistirilmesi gereken	Beklentim	0-25	MS Projec	Planlanan	Sinif
18	Buyuk olcude	kismen	50-75	Sistem analisti	Surecin nasil takip edilecegi	Iyi	Degisken musteri istekleri	Iyi	Katkisi be	50-75	MS Projec	Paraleldir	Sinif
19	Kismen	kismen	50-75	Sistem analisti	Surecin nasil takip edilecegi	Iyi	Surecin ve aktivitelerin iyi c	Gelistirilmesi gereken	Beklentim	25-50	Tahmini fi	Paraleldir	Sinif
20	Buyuk olcude	kismen	50-75	Sistem analisti	Surecin nasil takip edilecegi	Iyi	Degisken musteri istekleri	Gelistirilmesi gereken	Beklentim	0-25	MS Projec	Paraleldir	Sinif
21	Kismen	kullanilmamistir	50-75	Sistem analisti	Surecin nasil takip edilecegi	Iyi	Calisanlarin eksik performa	Iyi	Katkisi be	25-50	MS Projec	Planlanan	Sinif
22	Buyuk olcude	kismen	100	Sistem analisti	Calisanlarin eksik performansiiz	Kotu	Calisanlarin eksik performa	Kotu	Katkisi be	50-75	Tahmini fi	Planlanan	Sinif
23	Buyuk olcude	kismen	25-50	Veri tabani Tasarimcisi	Ilk gelistirilen buyuk proje	Orta	Calisanlarin eksik performa	Gelistirilmesi gereken	Beklentim	50-75	MS Projec	Planlanan	Sinif
24	Kismen	Buyuk olcude	25-50	Sistem analisti	Calisanlarin eksik performansiiz	Orta	Kaynak atamasi(Kisilerin isl	Iyi	Katkisi be	25-50	Tahmini fi	Paraleldir	Sinif
25	Buyuk olcude	kismen	50-75	Veri tabani Tasarimcisi	Calisanlarin eksik performansiiz	Orta	Degisken musteri istekleri	Gelistirilmesi gereken	Beklentim	25-50	Tahmini fi	Paraleldir	Ardisi
26	Kismen	kullanilmamistir		Sistem analisti	Ekip calismasi	Iyi	Surecin ve aktivitelerin iyi c	Gelistirilmesi gereken	Beklentim	50-75	MS Projec	Planlanan	Sinif
27	Kismen	Buyuk olcude	50-75	Veri tabani Tasarimcisi	Calisanlarin eksik performansiiz	Iyi	Proje alani ve konusu	Gelistirilmesi gereken	Katkisi be	25-50	MS Projec	Planlanan	Sinif
28	Kismen	kismen	50-75	Veri tabani Tasarimcisi	Surecin nasil takip edilecegi	Iyi	Calisanlarin eksik performa	Iyi	Beklentim	25-50	MS Projec	Paraleldir	Sinif
29	Buyuk olcude	kismen	50-75	Yazilimci	Ekip calismasi	Orta	Kaynak atamasi(Kisilerin isl	Gelistirilmesi gereken	Katkisi be	25-50	Tahmini fi	MS Projec	Ardisi

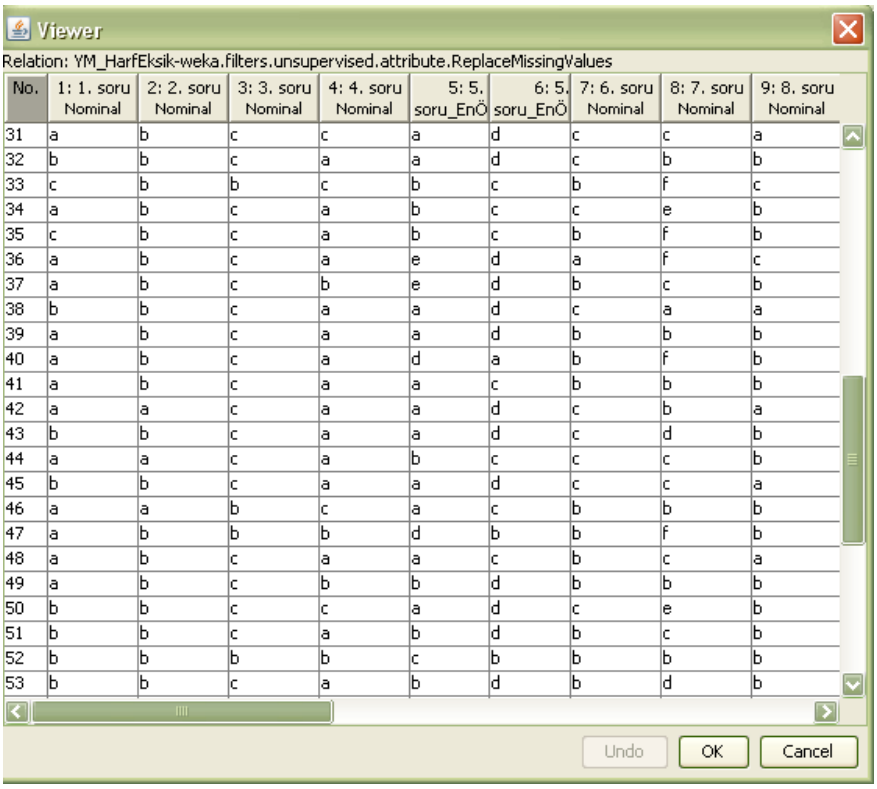
Şekil 4.1 YM ders anketlerinin içerik görüntüsü

Bu anketlere verilen cevapların şıklı hali ise Şekil 4.2'de görülmektedir. Ayrıca Veri madenciliği analizi için kullanılan Weka Programı csv, arff, c4.5 libsvm, xarff gibi formatları desteklemektedir. Bu nedenle veriler uygun formatlara çevrilerek dosyaların okunması sağlanmıştır. Böylelikle weka'da okunabilir hale gelen dosyalar veri madenciliği işlemlerini uygulamak için hazır hale getirilmiştir.



Şekil 4.2 YM ders anketlerine verilen cevapların şıklar ile gösterimi

Veri madenciliği algoritmasının uygulanacağı veri kümesinde Şekil 4.2'de de görüldüğü gibi bazı şıklar hiç işaretlenmemiş, bazı sorulara ise birden fazla cevap verilmiştir. Bu tür kayıp verilere sahip bir veritabanına uygulanabilecek yöntemlerden ilki kayıp verinin bulunduğu kaydı veritabanından çıkarmaktır. Eğer kayıp verili kayıt sayısı, toplam kayıt sayısına göre oldukça az ise bu kayıtların veritabanından çıkarılması mümkündür. Diğer taraftan kayıp veri sayısı yüksekse veya bu kayıtlara ait diğer değerler önemliyse kayıp değerlerin yerine genel bir sabit kullanılabilir ya da kayıp verilerin yerine tüm verilerin ortalama değeri kullanılabilir [67]. Kayıp verilerin yaratacağı sorunları ortadan kaldırmak için ilgili sorulara cevap atanması kayıp değer yerleştirme (Missing Value Replacement) işlemine göre yapılmıştır. Kayıp değer yerleştirme yaklaşımına göre cevabı boş olan soruya cevap olarak ilgili soru için en çok cevaplanan şıkkın ataması gerçekleştirilmiştir. Şekil 4.3 Kayıp verilere Replace Missing Values uygulandıktan sonraki durumunu göstermektedir.



No.	1: 1. soru Nominal	2: 2. soru Nominal	3: 3. soru Nominal	4: 4. soru Nominal	5: 5. soru_EnÖ	6: 5. soru_EnÖ	7: 6. soru Nominal	8: 7. soru Nominal	9: 8. soru Nominal
31	a	b	c	c	a	d	c	c	a
32	b	b	c	a	a	d	c	b	b
33	c	b	b	c	b	c	b	f	c
34	a	b	c	a	b	c	c	e	b
35	c	b	c	a	b	c	b	f	b
36	a	b	c	a	e	d	a	f	c
37	a	b	c	b	e	d	b	c	b
38	b	b	c	a	a	d	c	a	a
39	a	b	c	a	a	d	b	b	b
40	a	b	c	a	d	a	b	f	b
41	a	b	c	a	a	c	b	b	b
42	a	a	c	a	a	d	c	b	a
43	b	b	c	a	a	d	c	d	b
44	a	a	c	a	b	c	c	c	b
45	b	b	c	a	a	d	c	c	a
46	a	a	b	c	a	c	b	b	b
47	a	b	b	b	d	b	b	f	b
48	a	b	c	a	a	c	b	c	a
49	a	b	c	b	b	d	b	b	b
50	b	b	c	c	a	d	c	e	b
51	b	b	c	a	b	d	b	c	b
52	b	b	b	b	c	b	b	b	b
53	b	b	c	a	b	d	b	d	b

Şekil 4.3 YM ders anketine ReplaceMissingValues modülü uygulandıktan sonraki ekran görüntüsü

Sistem Analizi ve Tasarımı, Yazılım Mühendisliği ve Çevik Yazılım Geliştirme anketlerine Replace Missing Values modülü uygulandıktan sonra veri seti WEKA'da analizler için kullanıma hazır hale gelmiştir. Veriler ön işlemden geçirildikten sonra istatistiksel olarak değerlendirilip yorumlanmıştır. WEKA'da veri madenciliği yöntemlerinden olan sınıflandırma yöntemi kullanılarak sınıflandırma için en iyi algoritma keşfedilmiştir ve yine veri madenciliği yöntemlerinden biri olan birliktelik kuralı yöntemi kullanılarak sorular arasındaki bağlantılar gözlemlenmiştir. Uygulamada ilk önce eğitim evresi vardır, daha sonra üretilen kestirim modelini test etmek için 10 katlı çapraz onaylama kullanılmıştır.

n katlı çapraz geçerlilik: Veri sayısının sınırlı olması nedeniyle veri kümesinin bir bölümünün sınama amacıyla ayrılması mümkün olmadığında çapraz geçerlilik (cross validation) yönteminin uygulanması söz konusu olabilmektedir. Bu yöntemde veri kümesi rastgele iki eşit parçaya ayrılır. İlk aşamada bir parça üzerinde model eğitimi ve diğer parça üzerinde test işlemi; ikinci aşamada ise ikinci parça üzerinde model eğitimi

ve birinci parça üzerinde test işlemi yapılarak elde edilen hata oranlarının ortalaması kullanılır [68].

Verilerin örneğin 10 gruba ayrıldığı bu yöntemde, ilk aşamada birinci grup test, diğer gruplar öğrenim için kullanılır. Bu süreç her defasında bir grubun test, diğer grupların öğrenim amaçlı kullanılması ile sürdürülür. Çapraz geçerlilik süreci her bir alt örneğin test verisi olarak kullanılması için *n* kez tekrarlanır. Sonuçta elde edilen on hata oranının ortalaması, kurulan modelin tahmini hata oranı olacaktır.

Kullanılan algoritmaların başarılarını değerlendirmek için bazı kıstaslar değerlendirilerek algoritmaların başarıları karşılaştırılmıştır. Bu kıstaslar doğruluk değeri, ortalama mutlak hata, kök hata kareler ortalaması ve kappa istatistiğidir.

4.2 Veri Madenciliği Teknikleri

Sınıflandırma veri madenciliğinde sıkça kullanılan bir yöntem olup, veritabanındaki gizli örüntülerin ortaya çıkarılması yani verinin ortak özelliklerine göre ayrıştırılması için kullanılır. Sınıflandırma bir öğrenme algoritmasına dayanır, öncelikle veritabanının bir kısmı örnek veri kümesi olarak belirlenerek eğitim amacıyla kullanılır ve sınıflandırma kuralları oluşturulur, daha sonra bu kurallar yardımıyla yeni bir durum ortaya çıktığında nasıl karar verileceği belirlenir. Böylece hangi sınıfa ait olduğu bilinmeyen bir kayıt için bir sınıf belirlenebilir.

4.2.1 Karar Ağaçları ile Sınıflandırma

Karar ağaçlarını oluşturabilmek için belirli bir yol izlenir. Öncelikle veri arasından bir kısmı seçilerek eğitme işi yerine getirilir. Yani karar ağacının, belirli bir örneğe göre, yani eğitim kümesindeki veriye göre oluşturulması söz konusudur. Karar ağaçları oluşturulduktan sonra bu ağaçtan karar kuralları üretilir ve test verisi üzerinde denenir. Olumlu sonuç elde edilirse yeni gözlemleri sınıflandırmak için bu kurallar kullanılır. Diğer yöntemlerle karşılaştırıldığında karar ağaçlarının yapılandırılması ve anlaşılması daha kolaydır. Karar ağacı yapılarında, her düğüm bir nitelik üstünde gerçekleştirilen testi, her dal bu testin çıktısını, her yaprak düğüm ise sınıfları temsil eder. En üstteki

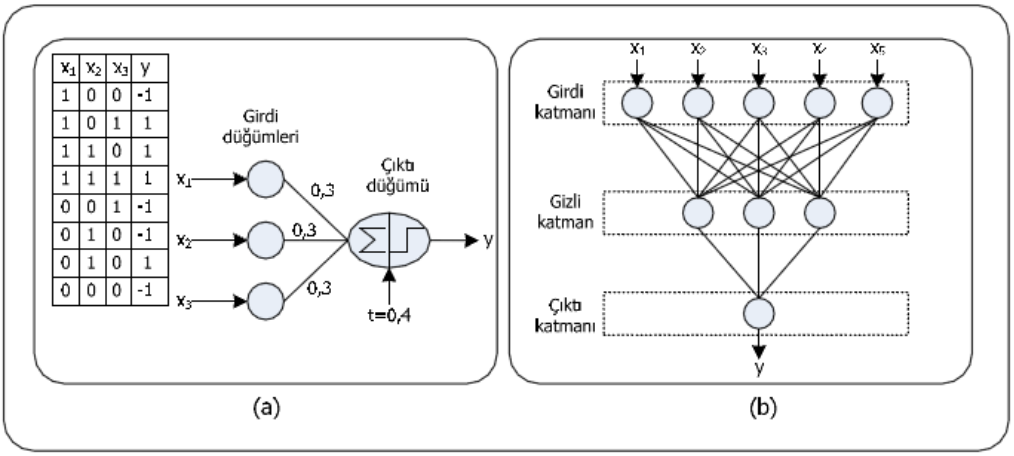
düğüm kök düğüm olarak adlandırılır. Karar ağaçları, kök düğümden yaprak düğüme doğru çalışır [69].

J48: J48 algoritması; C4.5 karar ağacı algoritmasının WEKA'ya uyarlanmış versiyonudur. Karar ağacı algoritmaları durumlar veya örnekler kümesiyle başlar ve yeni durumları sınıflandırabilmek için kullanılan ağaç veri yapısı yaratır. Her durum sayısal veya sembolik değer alabilen öznitelikler kümesi ile tanımlanır. Her bir eğitim durumu sınıfın adıyla ifade edilen etikettir. Karar ağacının her bir iç düğümü test içerir, testin sonuçları o düğümden hangi dalın takip edileceğine karar vermek için kullanılır. Yaprak düğümleri testler yerine sınıf etiketleri içerir. Sınıflandırma modu test durumunda, etiket yoksa yaprak düğümüne ulaşır, C4.5 orada bulunan etiketi kullanarak sınıflandırır. Bu alanların çoğu için, C4.5 tarafından üretilen ağaçlar küçük ve doğrudurlar, hızlı ve güvenilir sınıflandırıcılar meydana getirirler. Bu özellikler karar ağaçlarını sınıflandırma için değerli ve popüler araçlar yapar [70]. Böl ve fethet algoritması her yaprak tek sınıf durumu içerene kadar veya iki durumun farklı sınıflara ait olmalarına rağmen her bir öznitelik için aynı değerlere sahip olduğu daha fazla bölünmenin imkansız olduğu duruma kadar veriyi parçalara ayırır. Sonuç olarak, çelişkili durumlar içermiyorsa, karar ağacı tüm eğitim durumlarını doğru sınıflandıracaktır. Aşırı uyma, bazı eğitim durumları kümelerini alt bölümlere ayrılmaktan koruyan durdurma kriterleri ile veya üretildikten sonra karar ağacının bir kısım yapısını kaldırarak önlenebilir [72].

4.2.2 Yapay Sinir Ağları ile Sınıflandırma

Yapay sinir ağları ile sınıflandırmanın işleyiş yapısı, çıktı katmanına ulaşabilmek için ağırlıkların hesaplanmasına dayanır. Eğitim veri kümesi üzerinde hesaplanan ağırlıklar, test veri kümesi üzerinde kullanılarak öğrenmenin ne kadar gerçekleştiği belirlenir. Elde edilen ağırlıkların etkinliği doğrulanamazsa ağırlıklar üzerinde düzeltme ve yeniden hesaplama işlemleri gerçekleştirilir. Öğrenme süreci tamamlandığında ise ağırlıklar yardımıyla yeni bir verinin hangi sınıfa ait olduğu belirlenebilir. Yapay sinir ağlarında öğrenme süreci uzun sürse de oldukça duyarlı sınıflandırmalar yapabilmektedir. Şekil 4.4(a)'da yapay sinir ağlarının en basit hali Şekil 4.4(b)'de beş

değişken için çok katmanlı ileri yayılımlı bir yapay sinir ağı modelinin mimarisi verilmiştir.



Şekil 4.4 (a) Basit bir yapay sinir ağı modeli, (b) Çok katmanlı ileri yayımlı bir yapay sinir ağı örneği [72]

Multilayer Perceptron (MLP): MLP (çok katmanlı algılayıcı) örnekleri sınıflandırmak için geri yayılım (backpropagation) kullanan sınıflandırıcı içerir. Bu ağ elle yapılandırılabilir, algoritma ile yaratılabilir veya her ikisi de olabilir. Ağ ayrıca görüntülenebilir ve eğitim zamanı süresince modifiye edilebilir. Ağdaki düğümler sigmoidtir (bir şeyin haricinde; sınıf sayısal olduğu zaman çıkış düğümleri eşiksiz lineer birimler olur) [73].

SMO: Sequential Minimal Optimization kelimelerinin baş harflerinden oluşan SMO, esas itibariyle destek vektör (support vector) kullanan bir algoritmadır. Çok terimli (polynomial) kernel kullanarak destek vektör sınıflandırıcıyı eğitmek için SMO Algoritmasını uygular. Bu uygulama global olarak bütün kayıp değerleri yenisiyle değiştirir ve nominal öznitelikleri ikili olanlara dönüştürür. Ayrıca bütün öznitelikleri önceden tanımlanmış değerlerle normalize eder [74].

4.2.3 İstatistiğe Dayalı Algoritmalar

Regresyon ve korelasyona dayalı teknikler, Bayes teoremine dayalı sınıflama teknikleri ve veri özetlemede kullanılan tanımlayıcı istatistik yöntemleri veri madenciliğinde kullanılan başlıca tekniklerdir.

Regresyon ve korelasyon iki değişken arasındaki ilişkinin değerlendirilmesinde kullanılabilir. Regresyon genellikle noktalar kümesini bir eğriye uydurarak gelecek

değerleri geçmiş değerlere dayalı olarak tahmin etmede başvurulan bir tekniktir. Korelasyon ise iki değişken arasındaki benzerliği ölçmek ve sınıflama veya kümelemede benzerlik ölçümünü yapmak için kullanılabilmektedir.

Sınıflama problemlerini çözmede kullanılan istatistiğe dayalı bir teknik de Bayes teoremidir. Bayes teoreminden faydalanan Bayes sınıflayıcıları verilen bir örneğin özel bir sınıfa ait olma olasılığı gibi sınıf üyelik olasılıklarını tahmin edebilirler.

Naive Bayes (NB): Naive Bayes sınıflamada her bağımsız özelliğin katkısı analiz edilerek bir koşul olasılığı belirlenir. Sınıflama farklı özelliklerin etkileri birleştirilerek gerçekleştirilir. Algoritma test verisinden elde ettiği olasılıkları, sonucu bilinmeyen sınıfların etiketlenmesinde kullanır. Veri kümesindeki her özelliğin sınıflama problemine eşit katkıda bulunduğu ve katkıların birbirinden bağımsız olduğu varsayıldığında basit bir sınıflama olan NB sınıflayıcısı kullanılabilir.

Bayes Net: Bayes ağ öğrenmesi çeşitli araştırma algoritmalarını ve kalite ölçülerinin kullanımını kapsar. Bayesian ağ sıklıkla çok terimli (multinominal) verinin ayrık ve devamlı değişkenle modellenmesi için kullanılır. Bu algoritma, çözümün olasılık dağılımı kestirimi önermesini kullanır. Çok değişkenli veriyi modellemek için kullanılan dağılım teknikleri kestirimine yeni aday oluşturmak amacıyla Bayesian ağları kullanılır. Bu algoritma daha zor problem sınıflarını daha verimli ve güvenilir çözmek amacıyla var olan metodu geliştirir. Bayesian ağları modellenmiş verideki değişkenler arasındaki ilişkiyi şifreler. Bayesian ağları problemin yapısını sunar [75].

4.2.4 Mesafeye Dayalı Algoritmalar ile Sınıflandırma

Mevcut verilerin birbirlerine olan uzaklıkları ve benzerliklerine dayanan bu tür algoritmaların en yaygın kullanılanı *k*-en yakın komşu algoritmasıdır. Bu algoritmanın amacı, sınıfları belli bir örnek kümedeki gözlem değerlerini kullanarak yeni gözlemlerin hangi sınıfa ait olduğunu belirlemektir. Örnek kümedeki gözlemlerin her birinin, sonradan belirlenen bir gözlem değerine olan uzaklıklarının hesaplanması ve en küçük uzaklığa sahip *k* adet gözlemin seçilmesi esasına dayanır.

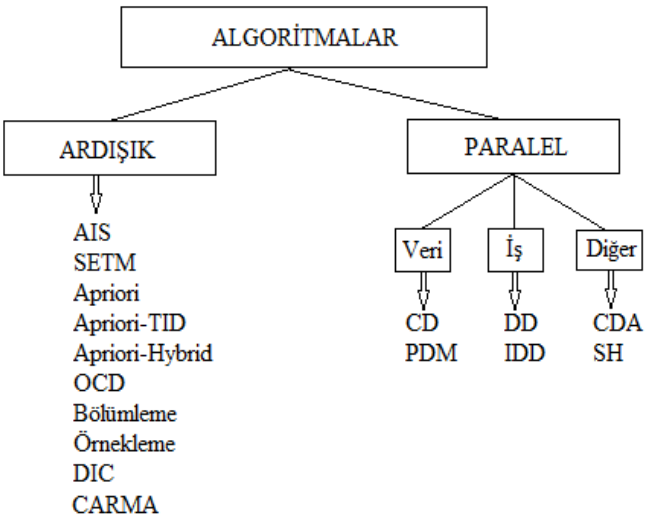
IBk: En yakın K-Komşu (K-Nearest Neighbors) algoritmasıdır. Bu algoritma sınıflandırma için kullanılır. K tabanlı komşuların uygun değerini çapraz doğrulama ile seçebilir. Ayrıca mesafe ağırlıklandırabilir [76].

KStar: K* örnek tabanlı bir sınıflandırıcıdır. Bazı benzerlik fonksiyonlarıyla belirlendiği gibi, eğitim örnekleriyle aynı olan sınıfa istinaden, test örneğinin sınıfıdır. Diğer örnek tabanlı öğrenenlerden entropi tabanlı mesafe fonksiyonu kullanması yönüyle farklıdır [77].

4.2.5 Birliktelik Kuralı

Veritabanı içinde yer alan kayıtların birbiriyle olan ilişkilerini inceleyerek, hangi olayların eş zamanlı olarak birlikte gerçekleşebileceklerini ortaya koymaya çalışan veri madenciliği yöntemleri bulunmaktadır. Bu ilişkilerin belirlenmesiyle birliktelik kuralları elde edilir [68].

Literatürde birliktelik kuralı çıkaran değişik algoritmalar bulunmaktadır. Apriori Algoritması, birliktelik kuralı çıkarım algoritmaları içerisinde en fazla bilinen algoritmadır [78]. Şekil 4.5'de Birliktelik kuralı algoritmaları gösterilmiştir.



Şekil 4.5 Birliktelik kuralı algoritmaları

Birliktelik kuralı yaklaşımlarında verilerin her biri parça (item) ve bu verileri birlikte oluşturduğu veri kümesi ise işlemleri, parça kümelerini, (transaction veya item set) oluşturmaktadır. İşlemler kümesi ise tüm veri kümesini tanımlamaktadır. İşlemlerde birlikte geçen veri parçalarının birbirleriyle olan meydana gelme ilişkisini çıkarmak için

kullanılan parametrelerden ilki destek (support) değeridir. A ve B veri parçalarının ilişkileri $A \Rightarrow B$ şeklinde gösterildiğinde destek değeri A ve B parçalarının tüm işlemlerde geçme sıklığının tüm işlem sayısına oranı olarak hesaplanmaktadır.

Destek (A $\Rightarrow$ B) = (A ve B'nin birlikte geçtiği işlem sayısı) / (toplam işlem sayısı)

Birliktelik kuralları aynı zamanda aşağıdaki gösterimdeki gibi çok boyutlu olarak da tanımlanabilmektedir.

$$\textit{Destek (A V C} \Rightarrow \textit{B)} = \frac{(A,B \text{ ve } C \text{ parçalarının birlikte geçtiği işlem sayısı})}{\text{toplam işlem sayısı}}$$

Birliktelik kurallarının çıkarımında destek değerinin yanı sıra güven (confidence) değeri de parçaların birlikteliklerinin çıkarılmasında önemli bir kriterdir. Güven değeri *A* veri parçasının geçtiği yerde B veri parçasının geçme sıklığının tüm işlem sayısına oranını vermektedir.

$$\textit{Güven (A V C} \Rightarrow \textit{B)} = \frac{(A,B \text{ ve } C \text{ parçalarının birlikte geçtiği işlem sayısı})}{A \text{ ve } B' \text{ninbirlikte geçtiği işlem sayısı}}$$

Birliktelik kuralında güven değeri çıkarımların gücünü verirken, destek değeri ise kurallardaki paternlerin frekansını vermektedir. Yüksek güven ve güçlü destek değerleri kullanılarak birbirleriyle bağlantılı olan yüksek ilişkili parçaların birliktelikleri çıkartılabilir. Bu nedenle güven ve destek değerlerinin önemli birliktelikleri vermesi için bu iki değerin de belirli eşik değerlerinden yüksek olması beklenmektedir. Yüksek güvenilirlik ve destek değerine sahip kurallara güçlü kurallar denir [79]. Güven değerinin % 100 olması durumunda, kural bütün veri analizlerinde doğrudur ve bu kurallara "kesin" denir.

Apriori Algoritması: Veri madenciliğinde kullanılan ve veri kümeleri veya veriler arasındaki ilişkiyi çıkarmak için geliştirilmiş algoritmadır. Birliktelik kuralının ilk aşaması için kullanılan Apriori Algoritması, sık geçen öğeler madenciliğinde kullanılan en popüler ve klasik algoritmadır [80]. Algoritmanın amacı, veri tabanında bulunan satırlar arasındaki bağlantıyı ortaya çıkarmaktır. Algoritmanın ismi, kendinden önceki çıkarımlara bağlı olduğu için, latince önce anlamına gelen "prior" kelimesinden gelmektedir.

Algoritma yapı olarak, aşağıdan yukarıya yaklaşımı kullanmakta olup, her seferinde tek bir elemanı incelemekte ve bu elemanla diğer adaylarla ilişkisini ortaya çıkarmaya çalışmaktadır. Ayrıca algoritmanın her eleman için çalışmasını, bir arama algoritmasına benzetmek mümkündür. Algoritma, bu anlamda yapısında olup, sanki adayları birer ağaç gibi düşünerek bu ağaç üzerinde arıyor kabul edilebilir. Ağaç yapısında, *k* elemanlı bir aday listesinden *k-1* elemana baktıktan sonra, alt frekans örüntüsü yetersiz olan elemanları budamakta ve kalan elemanların üzerinden arama yapmaya devam etmektedir [81].

4.3 Waikato Bilgi Analiz Platformu

WEKA (Waikato Environment for Knowledge Analyses), Waikato Üniversitesi tarafından geliştirilerek 1996'da ilk resmi sürümü yayınlanmış olan bir makine öğrenme ve veri madenciliği yazılımıdır. Akademik araştırmalar, eğitim ve endüstriyel uygulama alanlarında kullanım yeri olan WEKA, veri analizi ve tahminleyici modelleme için geliştirilmiş algoritma ve araçların görsel bir birleşimini içerir. Geliştirilen yazılımın temel avantajları geniş veri önişleme ve modelleme tekniklerine sahip olması, grafiksel kullanıcı arayüzü sayesinde kullanımının kolay olması ve Java programlama dili ile uygulandığından herhangi bir platformda kullanılabilmesi yani taşınabilir olmasıdır. Arff, Csv, C4.5 formatında bulunan dosyalar WEKA'da import edilebilir. Herhangi bir text dosyasındaki verileri WEKA ile işlemek olanaksızdır. Ayrıca Jdbc kullanılarak veritabanına bağlanıp burada da işlemler yapılabilir. WEKA 'nın içerisinde Veri İşleme, Veri sınıflandırma, Veri Kümeleme, Veri İlişkilendirme özellikleri mevcuttur [82].

WEKA makine öğrenmesi ve veri madenciliği için kapsamlıdır. WEKA'nın temel gücü bütün güncel makine öğrenmesi ve veri madenciliği yaklaşımlarının sınıflandırma alanında yatar. Gerileme, ilişki kuralları ve kümeleme algoritmaları da ayrıca uygulanmıştır.

4.4 Algoritmaların Karşılaştırılması

Karşılaştırmalar Ortalama Mutlak Hata (MAE) ve Kök Hata Kareler Ortalaması (RMSE) değerlerinin en küçük değer ilkesine göre yapıldı. Daha düşük bir hata oranı olan bir algoritma daha güçlü sınıflandırma yeteneğine sahiptir. Kestirim modelinin doğruluk

değeri de karşılaştırma için kullanıldı. Ayrıca algoritmaların karşılaştırılmasında kappa istatistiği oldukça önemlidir. Kappa istatistiği ne kadar yüksek olursa sınıflandırma başarımızın rastsal olmadığını gösterir. Yani anketlerde öğrencilerin büyük çoğunluğu aynı şıkka cevap vermişse doğruluk değeri yüksek çıkacaktır fakat kappa değeri bu dağılıma bağlı olmayıp ne kadar yüksek olursa o algoritmanın daha sağlıklı sınıflandırdığını gösterir. Bu nedenle kappa değeri en yüksek teknik en iyi teknik olarak seçildi.

4.4.1 Doğruluk Değeri

Sınıflandırmada başarı ölçütleri kullanılarak sınıflandırıcının test verileri üzerindeki etiket kararlarının başarı oranları ölçülmektedir. Sınıflandırmada başarı performans ölçütleri genellikle ikili sınıflama için önerilmiştir ve değerlendirme için hata matrisinde yer alan bilgiler kullanılır.

Çizelge 4.1'de hata matrisi yer almaktadır. Çizelgede gerçek sınıfı pozitif olan örneğin sınıflandırıcı tarafından pozitif olarak etiketlenmesi doğru pozitif (DP), gerçek sınıfı negatif olan bir örneğin sınıflandırıcı tarafından pozitif olarak etiketlenmesi yanlış pozitif (YP), gerçek sınıfı pozitif olan bir örneğin sınıflandırıcı tarafından negatif etiketlenmesi yanlış negatif (YN), gerçek sınıfı negatif olan örneğin sınıflandırıcı tarafından negatif olarak etiketlenmesi ise doğru negatif (DN) olarak gösterilmiştir.

Çizelge 4.1 Hata matrisi

		Tahmin Edilen Sınıf	
		Pozitif	Negatif
Gerçek Sınıf	Pozitif	Doğru Pozitif (DP)	Yanlış Negatif (YN)
	Negatif	Yanlış Pozitif (YP)	Doğru Negatif (DN)

Hata matrisi bir sınıflandırma modelinin performansının ne kadar iyi olduğunu belirlemek için gereken bilgiyi sağlamasına rağmen, bu bilgiyi tek bir sayı ile özetlemek farklı modellerin performanslarını karşılaştırabilmek için daha elverişlidir.

Bu, aşağıda da belirtildiği gibi doğruluk gibi bir performans ölçütü kullanılarak yapılabilir:

$$DoğrulukOranı = \frac{DP+DN}{DP+DY+DN+YN} \quad (4.1)$$

4.4.2 Ortalama Mutlak Hata (Mean Absolute Error)

Ortalama Mutlak Hata (MAE), tüm test durumlarında kestirimi yapılan ve gerçek değerin arasındaki farkın ortalamasıdır; ortalama kestirim hatasıdır [83]. MAE'nin hesaplanması için gerekli formül eşitlik (4.2)'de gösterildiği gibidir:

$$\frac{|a_1-c_1|+|a_2-c_2|+\ldots+|a_n-c_n|}{n} \tag{4.2}$$

Gerçek çıktının a, beklenen çıktının ise c olduğunu kabul edelim.

4.4.3 Kök Hata Kareler Ortalaması (Root Mean-Squared Error)

RMSE model veya kestirimci tarafından kestirimi yapılan değerler ve modellenen veya kestirimi yapılandan elde edilen gerçek değerler arasındaki farkın ölçüsü olarak sıklıkla kullanılır [83].

Hata Kareleri Ortalamasının (MSE: Mean Square Error) karekökü alınarak hesaplanır:

$$\sqrt{\frac{(a_1-c_1)^2+(a_2-c_2)^2+\ldots+(a_n-c_n)^2}{n}} \tag{4.3}$$

Hata kareleri ortalaması sayısal kestirimler için en çok kullanılan başarı ölçülerinden biridir. Bu değer, her hesaplanan değer ve onun karşılık gelen doğru değer arasındaki farkın karelerinin ortalaması alınarak hesaplanır. RMSE basitçe MSE'nin kareköküdür. RMSE hata değerini gerçek ve kestirilen değerdeki gibi aynı boyutta verir.

MAE ve RMSE her bir algoritma için hesaplanır.

- MAE ve RMSE kestirimler kümesindeki hatalardaki varyasyonu tanımlamak için beraber kullanılırlar.
- RMSE her zaman MAE'den büyüktür veya ona eşit olacaktır. Aralarındaki en büyük fark örnekteki özgün hatalardaki varyasyon en büyük olunca olur.
- Eğer RMSE, MAE'ye eşitse, tüm hatalar aynı önemdedir.
- RMSE ve MAE 0'dan ∞'a kadar değer alabilir.
- MAE ve RMSE negatif yönelimlidirler ve düşük değerler daha iyidir. Bu yüzden düşük MAE değerine sahip algoritma en iyi algoritma olarak değerlendirilir.

4.4.4 Kappa istatistiği (Cohen's kappa coefficient)

Değerlendiriciler arasındaki anlaşmanın istatistiksel ölçüsüdür ve herhangi bir ölçüm vakasının doğruluğunu değerlendirmek için kullanılır. Kappa istatistiği toplanan verilerin güvenilirliği ve bunların geçerliliğini birbirinden ayırt etmek için kullanılır.

Kappa = (Gözlenen anlaşma-Tesadüf anlaşma)/(1 – Tesadüf anlaşma)

Tam anlaşma durumunda kappa istatistiği değeri 1 olur. Kappa istatistiği değeri 1'e ne kadar yakınsa anlaşma o derece iyidir. Hiç bir anlaşmanın olmaması durumunda kappa istatistiği değeri sıfıra eşittir veya sıfırdan küçüktür.

4.5 Materyal

Yapılan anketlere Yıldız Teknik Üniversitesi, Bilgisayar mühendisliği bölümü, *Sistem Analizi ve Tasarımı* dersinden 86 öğrenci, *Yazılım Mühendisliği* dersinden 70 öğrenci ve Namık Kemal Üniversitesi, Bilgisayar mühendisliği bölümü, *Çevik Yazılım Geliştirme* dersinden 33 öğrenci yani toplamda 189 öğrenci katılmıştır. *Sistem Analizi ve Tasarımı*, *Yazılım Mühendisliği* ve *Çevik Yazılım Geliştirme* sırasıyla 2, 3 ve 4. sınıf dersleridir. Bu derslerdeki projeler bir dönemde yapılmaktadır. Proje gruplarındaki öğrenci sayıları derslere göre değişmekte olup 2 ile 6 kişi arasındadır. Önerilen proje konuları ve grup arkadaşları öğrenciler tarafından belirlenmektedir. Öğrencilere yapılan bu anketler, ders projeleriyle ilgili olup 20'şer sorudan oluşmaktadır. Bu anket sorularının bir kısmı öğrencilerin sosyal yeteneklerini ölçen sorulardan oluşurken bir kısmı da teknik yeteneklerini ölçen sorulardan oluşmaktadır.

Öğrencilerin sosyal yeteneklerini ölçen sorular; dokümantasyon, ekip içi uyum, projelerde zorlanılan alanlar ve öğrenci perspektifinden proje değerlendirilmesi gibi sorulardan oluşmaktadır. Teknik yeteneklerini ölçen sorular; geliştirme teknikleri (Veri Akış, Varlık İlişki (ER), Kullanıcı Senaryosu, Yapı Diyagramları) ve metodolojiler (Çevik, Şelale, Spiral, Uç (Extreme) Programlama) gibi öğrenci projelerinde uygulanması gerekenleri projelerinde ne denli uygulandığını ölçmeye dayalı sorulardan oluşmaktadır.

BÖLÜM 5

VERİ ANALİZİNİN GERÇEKLENMESİ

YM, SA ve ÇYG derslerinde öğrencilere ders kapsamında gerçekleştirdikleri projeleriyle ilgili yapılan anket sorularının bir kısmı Çizelge 5.1'de bulunmaktadır.

Çizelge 5.1 Öğrencilere yapılan anket soru örnekleri

Yazılım mühendisliği dersi anket soru örnekleri
Aşağıdaki diyagramlardan hangisi modelleme ve tasarımı ifade etmek için daha faydalı oldu?
Aşağıdaki diyagramlardan hangisi yazılım süreçlerinde ilk oluşturduğunuz diyagramdır?
Ardışıl Diyagram ve Sınıf Diyagramlarını projenin hangi aşamasında kullandınız?
Nesneye dayalı tasarımda anlaşabilirdik bakımından en uygun diyagram hangisidir?
Yazılım projenizde hangi yazılım geliştirme sürecini kullandınız?
Sistem analizi dersi anket soru örnekleri
ER Diyagram, Veri Akış Diyagramı ve Yapı Diyagramından uygulama süresince yararlanıldı mı?
Veritabanı oluşturulurken ER Diyagramdan yararlandınız mı?
Sadece veri akış diyagramı verilen bir yazılımın uygulanabilirliği hakkında ne düşünüyorsunuz?
Projenizde en az süre çalışan ekip üyeleri hangisidir?
MS Project ile belirlediğiniz yazılım geliştirme zamanı ile sürecin tamamlanmasından sonra elde edilen proje geliştirme zamanı birbiriyle paralellik göstermekte midir? İlk başta belirlenen proje süresi aşılmış mıdır?
Çevik yazılım geliştirme dersi anket soru örnekleri
Use Case Diyagramı ve Sınıf Diyagramlarını projenin hangi aşamasında kullandınız?
Daha önceki projelerinizi çevik yöntemler ile kıyasladığında aşağıdakilerden hangisi sizin için daha baskın bir farktır?
Test güdümlü yaklaşımı projenizde ne oranda kullandınız?
Sizce nesneye dayalı tasarımda anlaşılabilirlik bakımından en uygun diyagram hangisidir?
Çevik Yazılım dersi projesini diğer derslerdeki projelerle kıyasladığınızda hangi konuda daha fazla katkı sağlamıştır?

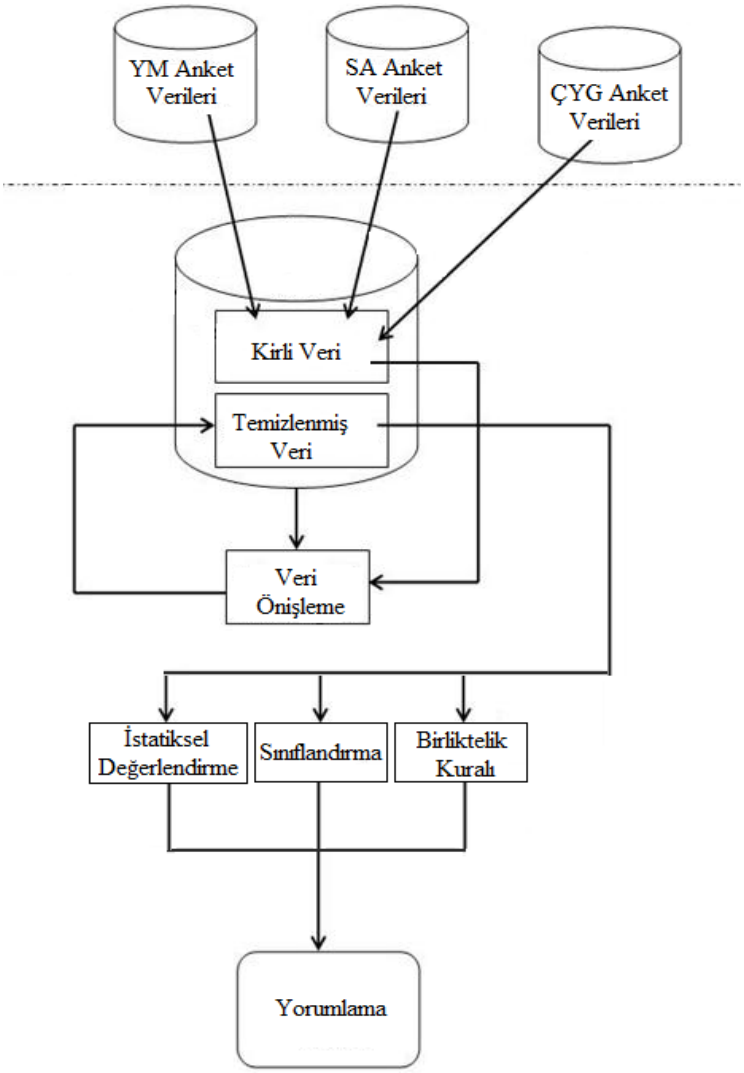
Çizelge 5.1 Devam

Ortak anket soru örnekleri
Yazılım geliştirici, sistem analiz ve tasarımından elde edilen süreç ve raporları ne ölçüde kullanmıştır?
Müşteri gözünden değerlendirirseniz; sürecin ilk başında istediğiniz yazılım ile süreç sonunda oluşturulan yazılım birbiriyle ne ölçüde örtüşmektedir? (Yüzde)
Yazılım geliştirici, uygulama geliştirme esnasında sistem analiz ve tasarım argümanlarının değişimini gerekli görmüş müdür?
Proje içi ekip uyumunuz ne ölçüdeydi?
Proje sonunda elde ettiğiniz ürüne müşteri gözünden bakarsanız nasıl değerlendirirsiniz?

Tez kapsamında, öncelikle YM, SA ve ÇYG ders projeleriyle ilgili öğrencilere yapılan anket verileri ile veri toplama ve depolama işlemleri yapılmıştır.

Veri depolama ortamı kullanılacak program dikkate alınarak csv ve arff formatında oluşturulmuştur. Daha sonra elde edilen bu veriler ile veri önişleme adımı uygulanmıştır. Veri önişleme aşamasında, verilerin veri madenciliği için hazır duruma getirilmesi için veri üzerinde veri tipi dönüşümü, gruplama, kayıp değerleri yönetme ve gürültülü verinin temizlenmesi gibi işlemlerin uygulandığı aşamadır. Gürültülü veri de veri önişleme aşamasında veri kümesinden temizlenir. Gürültülü veri, veri kümesi içinde yer alan ama veri madenciliği uygulamasında kullanılmayacak ve bir anlam içermeyen verilerdir [84].

Elde ettiğimiz temiz veriyi istatistiksel değerlendirme, sınıflandırma ve birliktelik kurallarında kullanıp analiz sonuçları yorumlanmıştır. Gerçekleştirilen işlem adımları Şekil 5.1'de görülmektedir.



Şekil 5.1 Gerçekleştirilen işlem adımları

BÖLÜM 6

BULGULARIN DEĞERLENDİRİLMESİ

Sistem Analizi ve Tasarımı, *Yazılım Mühendisliği* ve *Çevik Yazılım Geliştirme* ders anketlerindeki sosyal yetenek soruları hem derslerin karşılaştırmalı hali olarak hem de ortak sorular birleştirilerek bütün olarak değerlendirilmiştir. Her derste kullanılan diyagramlar, metotlar ve yöntemler farklı olduğu için teknik yetenek soruları ders bazında analiz edilmiştir. Anket soruları veri madenciliği yöntemlerinden olan sınıflandırma yöntemi ile değerlendirilip anketlerin sınıflandırılması için en başarılı olan algoritma belirlenmiştir. Ayrıca Birliktelik kuralı kullanılarak sorular arasındaki bağlantılar gözlemlenmiştir.

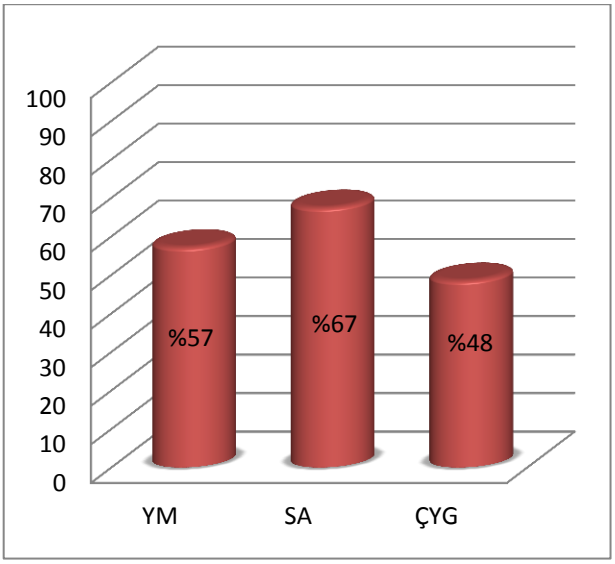
6.1 Anketlerin Değerlendirilmesi

Anket soruları öğrencilerin sosyal ve teknik yeteneklerini ölçen sorulardan oluşmaktadır. Bu bölümde, öğrencilerin anketlere verdikleri cevaplar doğrultusunda sorular incelenerek elde edilen sonuçlar ve yorumlar sosyal ve teknik yetenekler başlıkları altında değerlendirilmiştir. Bu değerlendirmelerin bir kısmı ortak soruların birleştirilmesiyle elde edilen veriler değerlendirilirken bir kısmı da ders bazında değerlendirilmiştir.

6.1.1 Tüm Ders Projelerinin Sosyal Yetenekler Açısından Değerlendirilmesi

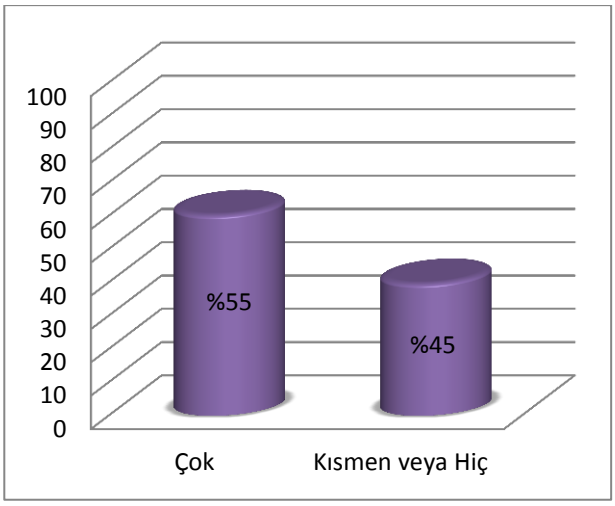
Şekil 6.1 Yazılım geliştiricinin sistem analiz ve tasarımından elde edilen süreç ve raporları ne ölçüde kullandığını ders bazında göstermektedir. ÇYG projesi test tabanlı bir proje olduğu için raporlamaya fazla ihtiyaç duyulmaz. SA dersinin yüzdesinin YM dersinden daha yüksek çıkmasının sebebi YM ders projesinde süreç kısadır, daha az

modül gerçekleştirilir, diyagramlar artar ve raporlama azalır. Fakat SA dersinde yapılan projeler, YM dersinde yapılan projelere göre daha kapsamlı ve tüm proje gerçekleştirilmektedir. Bu nedenle de süreç ve raporlamanın en fazla kullanıldığı ders SA dersidir. Elde edilen yüzdeleri ders bazında incelediğimiz zaman sonuçlar tutarlı çıkmıştır.



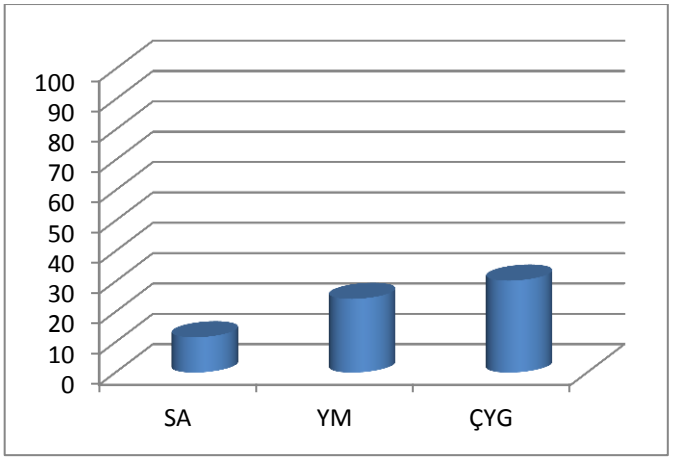
Şekil 6.1 Ders bazında yazılım geliştiricinin sistem analiz ve tasarımından elde edilen süreç ve raporları ne ölçüde kullandığının dağılımı

Şekil 6.2 Yazılım geliştiricinin sistem analiz ve tasarımından elde edilen süreç ve raporların ne ölçüde kullandığı tüm dersler bazında göstermektedir. Öğrencilerin %55'i süreç ve raporlamayı çok kullanmıştır. Süreç ve rapor kullanımı önemli olup, raporlamanın iyi yapılmaması durumunda çeşitli sorunlar çıkmaktadır. Eksik veya yanlış dokümantasyon, yazılım geliştirmede verimlilik ve kalite bakımından önemli bir eksikliğe yol açabilir [85]. Yapılan çalışmada süre kısıtlılığından dolayı geliştiricilerin dokümantasyondan çok geliştirmeye yöneldikleri gözlemlenmiştir. Yazılım mühendisliği aktiviteleri doküman hazırlayabilme yeteneği gerektirmektedir [86], [87]. Ancak yazılım mühendisleri bu anlamda son derece zayıftır [57]. Halbuki bu alanda çalışan kişiler sıklıkla bu yeteneklerin bir grup çalışmasında en az teknik yetenekler kadar önemli olduğunu vurgulamaktadır [88]. Bu nedenlerden dolayı anket sonuçlarından elde ettiğimiz %55 oranı oldukça düşüktür.



Şekil 6.2 Yazılım geliştiricinin sistem analiz ve tasarımından elde edilen süreç ve raporları ne ölçüde kullandığının dağılımı

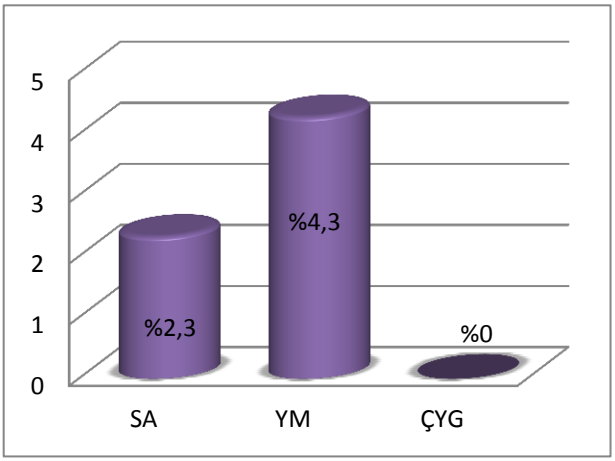
Ders bazında yazılım geliştiricinin argüman değişiminin çok olduğu oranlar Şekil 6.3'de gösterilmektedir. ÇYG ders projesinde argüman değişimine daha fazla ihtiyaç duyulmaktadır. Çünkü ÇYG ders projeleri müşteri ile görüşmelere ve müşterinin istediği doğrultuda sürekli değişimlere açıktır [89]. ÇYG ders projesinde argümanların daha fazla değişmesinin sebebi sürekli değişen müşteri isteklerinden dolayı proje üzerinde sıkça değişikliklere gidilmesidir. Bu nedenle de argüman değişikliği diğer projelere oranda daha fazla olmaktadır.



Şekil 6.3 Ders bazında yazılım geliştiricinin argüman değişiminin çok olduğu oranlar

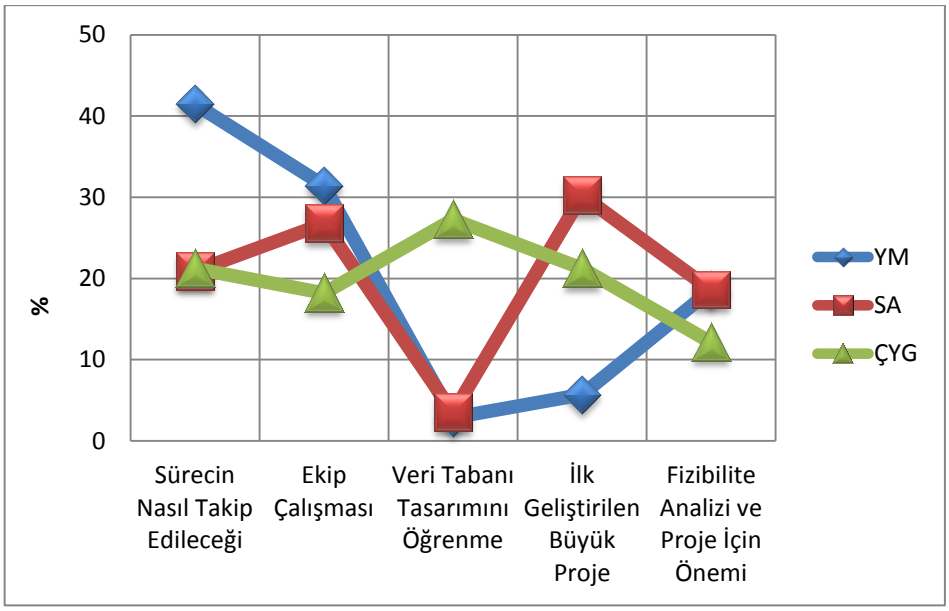
Şekil 6.4 Müşterinin sürecin ilk başında istediği yazılım ile süreç sonunda oluşturulan yazılımın birbiriyle örtüşmeme yüzdesini ders bazında göstermektedir. YM ve SA dersler projelerinde müşterinin sürecin ilk başında istediği yazılım ile elde edilen

ürünün birbiriyle örtüşme yüzdesi 0-25 arasında yani neredeyse müşterinin istediği gibi bir ürün olmaması mümkündür. Fakat ÇYG ders projesinde müşteri süreç sonunda bir sürprizle karşılaşıp farklı bir ürün görmesi mümkün değildir. Bunun sebebi çevik yazılım manifestolarından biri olan müşteri ile işbirliğidir. Yani müşteri ile geliştirici aynı takım içindedir [90]. Bu nedenle müşteri projenin her adımında istekleri doğrultusunda projeyi yönlendirmekte ve süreç sonunda istediği gibi bir ürün elde etmektedir. SA ders projelerinin müşteri isteğiyle uyuşma oranının, YM ders projesinden daha fazla olmasının sebebi; SA dersinde tüm proje gerçekleştirilirken YM ders projelerinde süreç kısadır, daha az modül gerçekleştirilir ve tasarım yapılmadığından dolayı müşteri isteğiyle en fazla örtüşmeyen projeler YM dersinde ortaya çıkmaktadır.



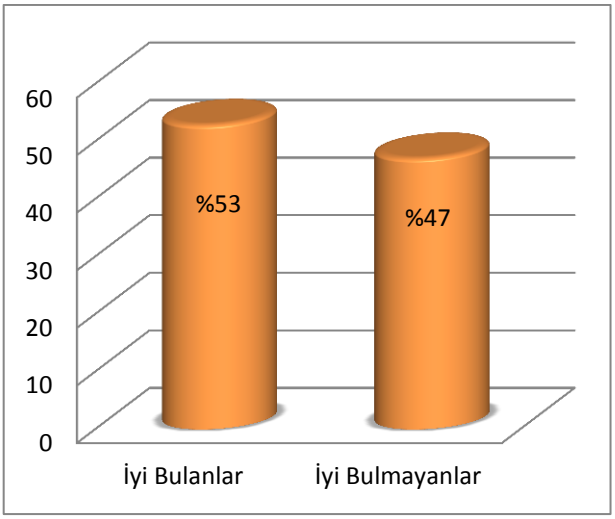
Şekil 6.4 Müşterinin sürecin ilk başında istediği yazılım ile süreç sonunda oluşturulan yazılımın birbiriyle örtüşmeme oranları

Şekil 6.5 Geliştirilen yazılım projesinin öğrencilere en büyük katkısını ders bazında göstermektedir. Veriler analiz edildiğinde en yüksek çıkan cevaplar; SA projesi için ilk geliştirilen proje olduğu, YM projesi için sürecin nasıl takip edileceği, ÇYG projesi için ise veritabanı tasarımını öğrenme olarak çıkmıştır. SA dersi 2. sınıf, YM dersinin 3. sınıf, ÇYG dersinin 4. sınıfta alındığı için öğrenciler SA dersi projesinde öğrenciler ilk olarak büyük bir proje geliştirmektedirler. *Waterfall, spiral, unified process, agile* gibi yazılım süreç ve modelleri ilk olarak YM dersinde öğrenilip geliştirilmektedir bu nedenle de YM ders projesinde öğrenciler sürecin nasıl takip edileceğini öğrenmektedirler ve bu da onlara en fazla katkıyı sağlamıştır. ÇYG dersinde öğrenciler projede en fazla veritabanı tasarımını öğrenmede katkı sağladığını söylemişlerdir ancak verdikleri bu cevap projeleriyle örtüşmemektedir.



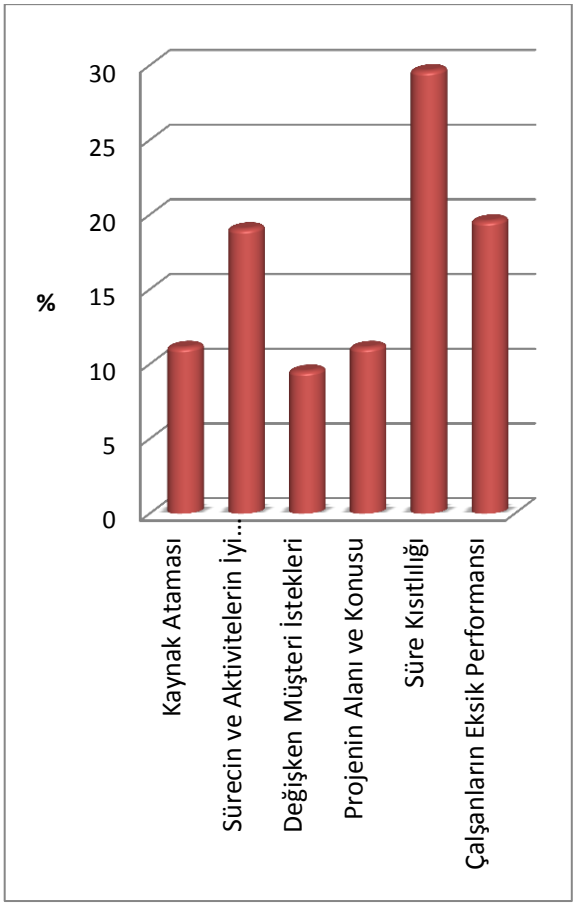
Şekil 6.5 Geliştirilen yazılım projesinin öğrencilere sağladığı katkılar

Şekil 6.6 Proje içi ekip uyumunu tüm dersler bazında göstermektedir. Öğrencilerin %47'si yani yaklaşık yarısı ekip uyumunu iyi bulmamaktadır. Bu oran oldukça yüksek bir orandır. Çünkü ekip içi uyum sektörde önemli bir yerde olup başarısızlıkların nedenlerinin başında gelmektedir. Yapılan çalışmalar yeni mezun öğrencilerin iletişim ve işbirliğinin iyi olmadığını göstermekte olup bunların iyileştirilmesinin gerektiği belirtilmektedir [39], [40]. Ayrıca birçok çalışmada ise yazılım mühendislerinin iyi bir grup üyesi olabilmeleri için eğitilmelerinin gerekliliği üzerinde durulmuştur [46], [86], [87].



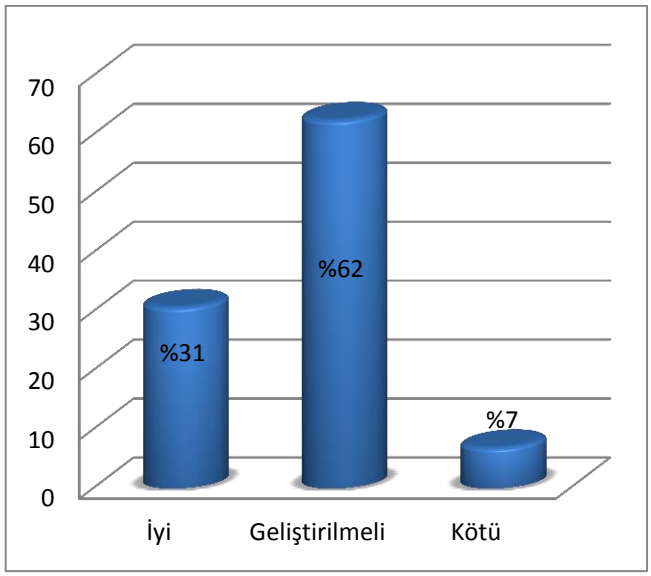
Şekil 6.6 Proje içi ekip uyum oranları

Şekil 6.7 Öğrencilerin proje tasarım ve analiz sürecinde en çok hangi alanda zorlandıklarını göstermektedir. Öğrenciler en çok süre kısıtlılığından dolayı daha sonra ise çalışanların eksik performansı yani yukarıda bahsedilen ekip içi uyumun kötü olmasından dolayı zorluk çekmektedir. Sonucun böyle çıkması oldukça tutarlı bir durumdur. Çünkü projeler bir dönemde yapılmakta ve aynı dönem içinde öğrencilerin başka derslerden de projeleri ve sınavları bulunmaktadır. Begel ve Simon tarafından yapılan çalışmada [40], gözlemler sonucunda pek çok yerde yazılımlardaki hataları düzeltmeye yönelim olduğunu ama bunun için vakit olmamasının dile getirilmesidir. Bu kanı özellikle büyük ve karmaşık geliştirme ortamlarında çok tehlikelidir. Ayrıca bir kuruluşta, bir yazılım ürününün geliştirilmesini koordine etmek için konfigürasyon yönetimi aracı kullanmak istediklerinde sorunun yeni teknolojiye uyum sağlamakta olmadığını, geliştiricilerin süre kısıtlılığından dolayı dokümantasyondan çok geliştirmeye odaklanmalarıdır [91].



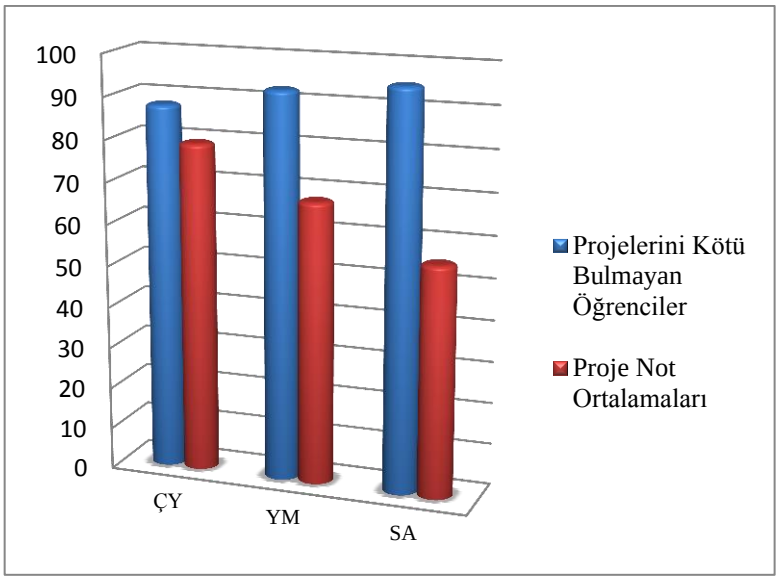
Şekil 6.7 Proje tasarım ve analiz sürecinde zorlanılan alanlar

Şekil 6.8 Proje sonunda elde edilen ürünün öğrenciler açısından değerlendirme sonuçlarını tüm dersler bazında göstermektedir. Sonuçlar incelendiğinde öğrencilerin sadece %31'i projesini iyi bulmuştur. 1995 yılında ABD'de yapılan çalışmada yazılım projelerinin ortalama %16'sının başarılı, %31'inin tamamlanamadan iptal edildiğini, tamamlanan projelerin ise yarıdan fazlasının tahmin edilen bütçenin %189 üzerinde bir maliyet ile tamamlandığını göstermektedir [24]. 1999 yılında yapılan araştırmada bu projelerin yaklaşık %75'inin ya zamanında tamamlanamadığını ya da iptal edildiğini göstermektedir [25]. 2009 yılında aynı grup tarafından yapılan çalışmada, başarılı projelerin ortalaması %32'ye çıkmasına rağmen halen çok düşük seviyededir. 2009 yılında başarısız projelerin oranı %24, değiştirilmiş projelerin oranı ise %44'tür [26]. Bizim çalışmamız 2012 yılında yapılmış olup öğrencilerin %31'i projelerini başarılı bulmuştur. Elde ettiğimiz sonuçlar daha önce yapılan analiz sonuçlarına paraleldir ancak geçen zamana karşın ciddi bir artış gözlemlenmemektedir. Elde edilen diğer sonuçlar da düşünüldüğünde proje başarısının düşük çıkması şaşırtıcı bir durum değildir. Çünkü süre kısıtlılığı, çalışanların eksik çalışması ve ekip içi uyumun iyi olmamasından dolayı süreç sonunda tatmin edici ürün elde edilmemektedir. Nitekim Nakakoji ve arkadaşları tarafından yapılan bir çalışmada yazılım geliştirme süresince yazılımcıların toplum içinde birbiriyle ilişkilerinin nasıl olduğu bütün yazılım sisteminin kalitesini etkilediği belirtilmiştir [92].



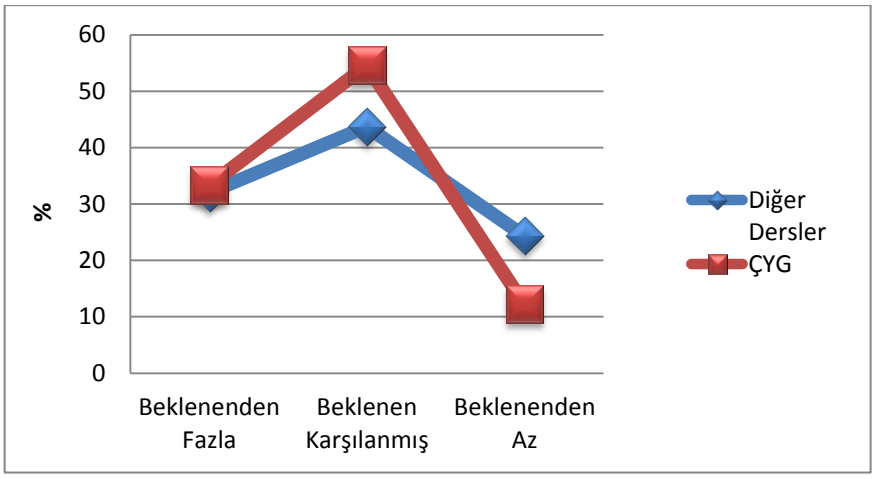
Şekil 6.8 Proje sonunda elde edilen ürünün öğrenciler açısından değerlendirilmesi

Şekil 6.9 Proje sonunda elde edilen ürünün öğrenci perspektifinden değerlendirilmesi ile projelerden alınan not ortalamalarının karşılaştırılmasını ders bazında göstermektedir. Sonuçlar karşılaştırıldığında proje sonunda elde edilen ürünü beğenenlerin not ortalamalarının daha düşük, proje sonunda elde edilen ürünü beğenmeyenlerin ise not ortalamalarının daha yüksek olduğu görülmektedir. Bunun sebebi dokümantasyon ve testlerin yetersiz yapılmasıdır. Çünkü dokümantasyonun yapılmaması veya eksik yapılması durumunda yeteri kadar geri bildirim alamamaktadırlar. Bu nedenle de yapılan hataları göremeyerek, yaptıklarının tamamının doğru olduğunu düşünmektedirler. Eğer yeteri kadar test ve dokümantasyon yapılsaydı hatalarını daha iyi göreceklerinden kendilerine göre projeleri değerlendirdiklerinde elde edilecek sonuçlar gerçeğe daha yakın çıkabilirdi.



Şekil 6.9 Proje sonunda elde edilen ürünün öğrenci perspektifinden değerlendirilmesi ile projelerin not ortalamalarının karşılaştırılması

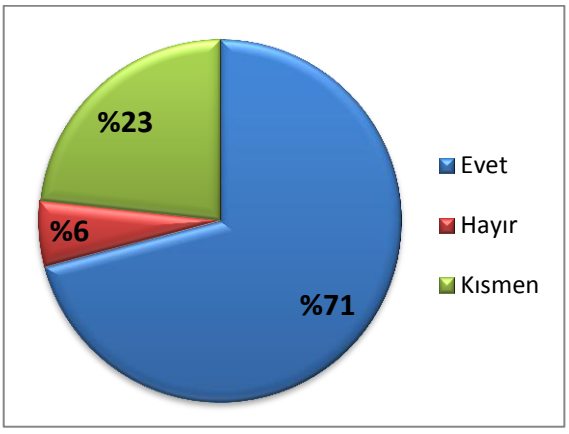
Şekil 6.10 ÇYG ders projesinin diğer ders projelerine göre yazılım geliştirme sürecinde öğrencilere ne ölçüde katkı sağladığını göstermektedir. Şekilde de görüldüğü gibi ÇYG ders projesini diğer derslerdeki projelerin katkısı ile kıyasladığımızda ÇYG ders projesinin katkısı hem daha fazla olmuştur hem de katkısının beklenenden az olma oranı yarıya düşmüştür. ÇYG' de beklentilerin daha fazla karşılanmış olmasının sonucu olarak öğrencilerin not ortalamalarının diğer derslerden daha yüksek olmasına sebep olmuştur.



Şekil 6.10 Projenin yazılım geliştirme sürecinde sağladığı katkı

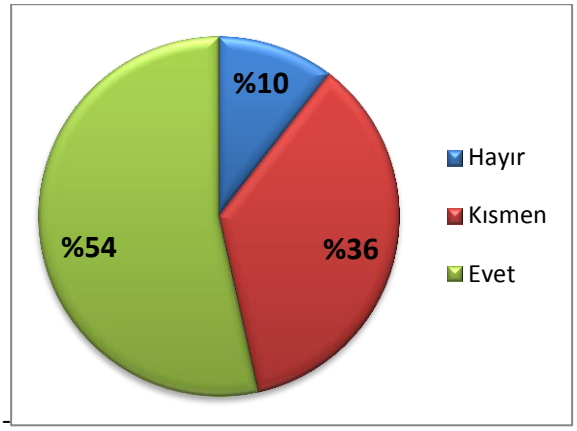
6.1.2 Sistem Analizi ve Tasarımı Ders Projelerinin Teknik Yetenekler Açısından Değerlendirilmesi

SA ders projesinde öğrencilerin Varlık İlişki (ER), Veri Akış ve Yapı Diyagramlarından uygulama süresince yararlanılma yüzdeleri Şekil 6.11'de gösterilmiştir. Veri akış diyagramları ve ER diyagramları yazılım mühendisliğinin ana gösterimleridir. Yazılım mühendisleri bu diyagramları bilgi sistemini anlamak, geliştirmek ve değerlendirmek için kullanırlar [92]. Bu nedenle de bu diyagramlar projenin büyük bir kısmında uygulanmaktadır. Anket sorularına verilen cevaplar değerlendirildiğinde öğrencilerin %71'i projesinde Veri Akış, ER ve Yapı diyagramlarından yararlanmış, %23'ü bu diyagramlardan kısmen yaralanmış ve %6'sı bu diyagramlardan yaralanmamıştır. Yani projede bu diyagramlardan yararlanmayan öğrenciler sadece %6 oranındadır.



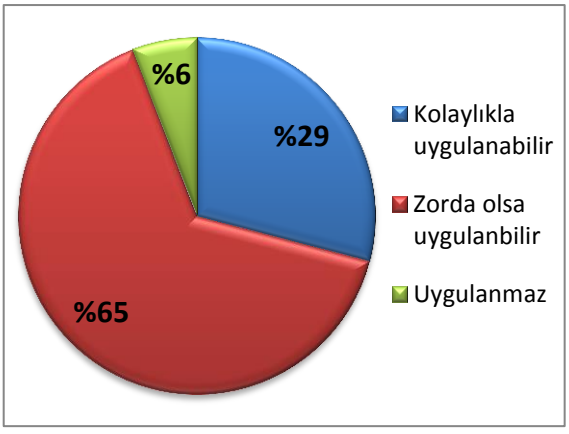
Şekil 6.11 SA Ders projesinde ER, veri akış ve yapı diyagramlarından uygulama süresince yararlanma oranları

SA ders projesinde öğrencilerin veritabanı oluştururken ER diyagramından yararlanma yüzdeleri Şekil 6.12'de gösterilmiştir. Öğrencilerin verdikleri cevaplar değerlendirildiğinde öğrencilerin sadece %10'u projesinde veritabanı oluştururken ER diyagramından yararlanmamıştır. Nitekim veritabanı oluşturulurken çoğunlukla kullanılan diyagram ER diyagramıdır [94].



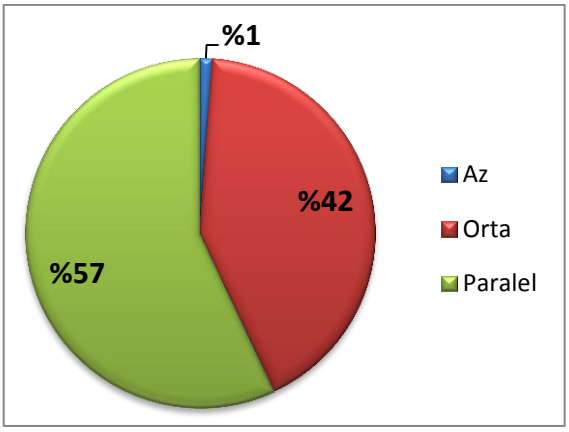
Şekil 6.12 SA Ders projesinde veritabanı oluştururken ER diyagramından yararlanma oranları

SA ders projesinde sadece veri akış diyagramı verilen bir yazılımın uygulanabilirliği hakkındaki öğrencilerin düşünceleri Şekil 6.13'de gösterilmiştir. Uygulama gerçekleştireceğimiz sistem hakkında önceden bilgimiz olmamasına rağmen veri akış diyagramları bize uygulama sürecindeki işlemler ve bu işlemlerle veri kaynakları arasındaki bilgi akışını anlattığı için zorda olsa uygulanabilir sonucunun % 65 çıkması tutarlıdır.



Şekil 6.13 SA Ders projesinde sadece veri akış diyagramı verilen bir yazılımın uygulanabilirliği hakkındaki düşünceler

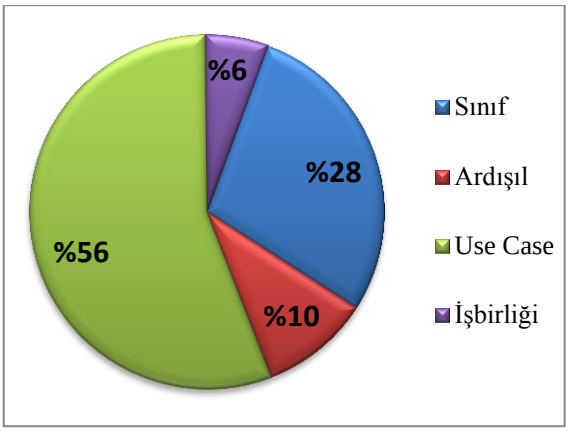
SA ders projesinde tasarlanan ile geliştirilen projenin varlık, modül ilişkileri ve veri akışlarının birbirlerine paralellik yüzdeleri Şekil 6.14'te gösterilmiştir. SA ders projelerinde uygulamanın kod geliştirme aşaması da yapılmaktadır. Öğrenciler veri akış diyagramını anlayarak çizimini gerçekleştirdiğinde, programla paralellik gösterdiği takip edilebilmiştir. Bu sonuç Şekil 6.13'te görülen sistem hakkında bilgi verilmeden sadece veri akış diyagramı verilen bir yazılımın zor da olsa uygulanabildiğini desteklemektedir.



Şekil 6.14 SA Ders projesinde tasarlanan ile geliştirilen projenin varlık, modül ilişkileri ve veri akışlarının birbirlerine paralellik oranları

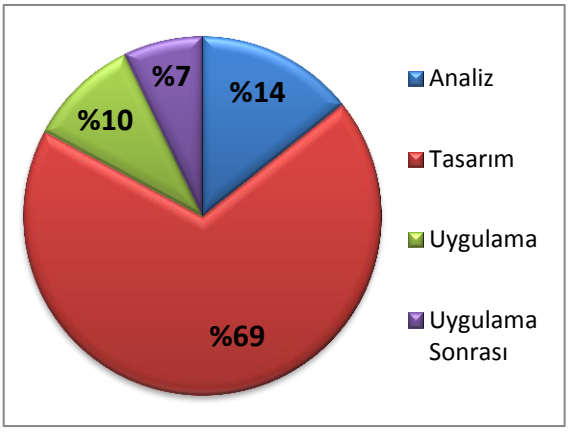
6.1.3 Yazılım Mühendisliği Ders Projelerinin Teknik Yetenekler Açısından Değerlendirilmesi

YM ders projesinde öğrencilerin yazılım süreçlerinde ilk oluşturdukları diyagramların yüzdeleri Şekil 6.15'de gösterilmiştir. Bir yazılım mühendisinin yaptığı ilk şey kullanıcı senaryosu (Use Case) diyagramlarını çizmektir [95]. Ayrıca Fowler ve Scott da işinde ilk olarak kullanıcı senaryosu diyagramlarını oluşturduğunu belirterek, kullanıcı senaryosu diyagramlarının kullanıcılara yardımcı olduğunu ortaya çıkarmıştır [96]. Anketlere verilen cevaplar değerlendirildiğinde öğrencilerin yarısından fazlası (%56), yazılım süreçlerinde kullanıcı senaryosu diyagramlarını oluşturmuştur.



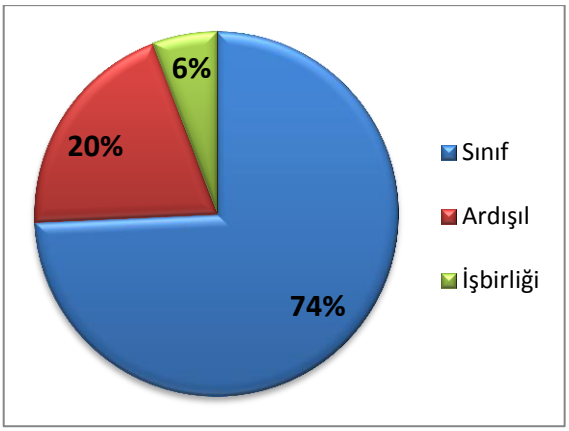
Şekil 6.15 YM ders projesinde öğrencilerin yazılım süreçlerinde ilk oluşturdukları diyagramlar

Şekil 6.16 YM ders projesinde öğrencilerin ardışıl ve sınıf diyagramlarını projenin hangi sürecinde kullanıldıklarını göstermektedir. Ardışıl ve sınıf diyagramları tasarım aşamasında kullanılır [97]. Nitekim anketler değerlendirildiğinde öğrencilerin %69'u projesinde ardışıl ve sınıf diyagramını tasarım aşamasında kullanmıştır.



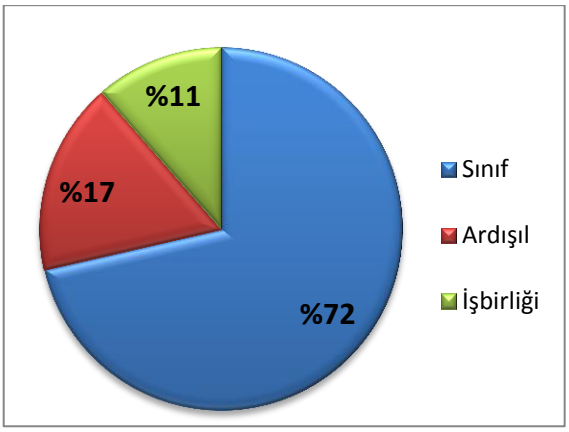
Şekil 6.16 YM ders projesinde ardışıl ve sınıf diyagramının kullanım aşaması

YM ders projesinde öğrenciler açısından anlaşılabilirlik bakımından en uygun diyagramlar Şekil 6.17'de gösterilmektedir. Sınıf diyagramı en temel diyagramdır ve sistem ile statik sınıf arasındaki varolan sınıf, arayüz ve öznitelik ilişkilerini gösterir [98]. Bu diyagramlarda nesneler arasındaki iletişimin dinamik yapısı görünmez [89]. Sınıf diyagramları kendi başlarına son derece yalın yapılardır [99]. Sınıf diyagramının bu yalın yapısından dolayı anlaşılabilirliği diğer diyagramlarından daha fazladır. Nitekim öğrencilerin %74'ü sınıf diyagramın analaşılabilirlik bakımından en uygun diyagram olduğunu düşünmektedirler.



Şekil 6.17 YM ders projesinde öğrenciler açısından anlaşılabilirlik bakımından en uygun diyagram oranları

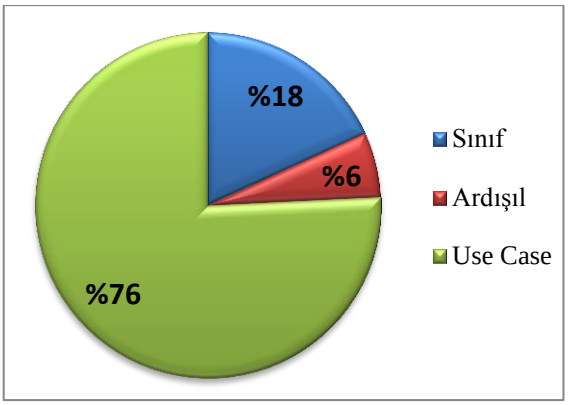
YM ders projesinde öğrenciler için modelleme ve tasarımı ifade etmede en faydalı bulunan diyagramların yüzdeleri Şekil 6.18'de gösterilmektedir. Sınıf diyagram object-oriented yöntemlerinde merkezi bir konumdadır, ayrıca modelleme ve tasarımı ifade etmek için sınıf diyagramı kullanılır [96]. Öğrencilerin %72 gibi büyük bir çoğunluğu modelleme ve tasarımı ifade etmede en faydalı diyagramın sınıf diyagram olduğunu düşünmektedirler.



Şekil 6.18 YM ders projesinde modelleme ve tasarımı ifade etmede faydalı bulunan diyagram oranları

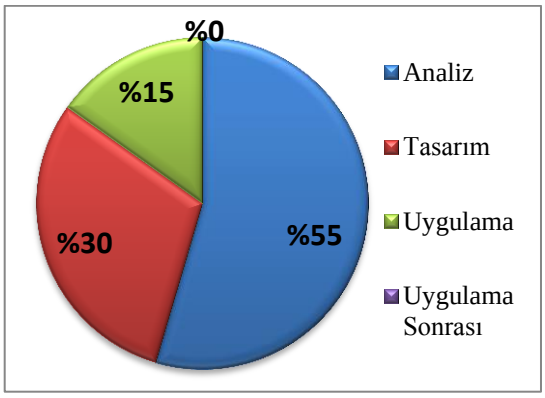
6.1.4 Çevik Yazılım Geliştirme Ders Projelerinin Teknik Yetenekler Açısından Değerlendirilmesi

ÇYG ders projesinde öğrencilerin yazılım süreçlerinde ilk oluşturdukları diyagramların yüzdeleri Şekil 6.19'da gösterilmiştir. Daha öncede belirttiğimiz gibi bir yazılım mühendisinin yaptığı ilk şey Use Case diyagramlarını çizmektir ve öğrencilerin %76'sı projesinde ilk kullanıcı senaryosu diyagramlarını oluşturmuştur.



Şekil 6.19 ÇYG ders projesinde öğrencilerin yazılım süreçlerinde ilk oluşturdukları diyagram oranları

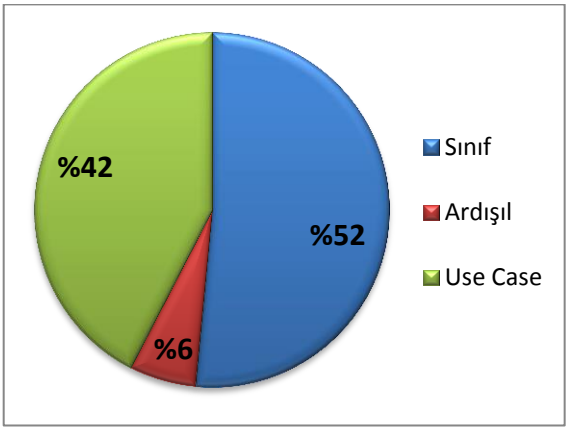
Şekil 6.20 ÇYG ders projesinde öğrencilerin kullanıcı senaryosu ve sınıf diyagramlarını projenin hangi sürecinde kullanıldıklarını göstermektedir. Kullanıcı senaryosu diyagramları ve kavramsal sınıf diyagramları projenin analiz aşamasında, tasarım sınıf diyagramları ise projenin tasarım aşamasında kullanılır [101]. Nitekim anket soruları değerlendirildiğinde ÇYG ders projesinde öğrencilerin %55 analiz aşamasında, %30'u tasarım aşamasında bu diyagramları kullanmıştır. Bu da diyagramların projenin hangi aşamasında kullandıklarının bilincinde olduklarını göstermektedir.



Şekil 6.20 ÇYG ders projesinde kullanıcı senaryosu ve sınıf diyagramının kullanım aşaması oranları

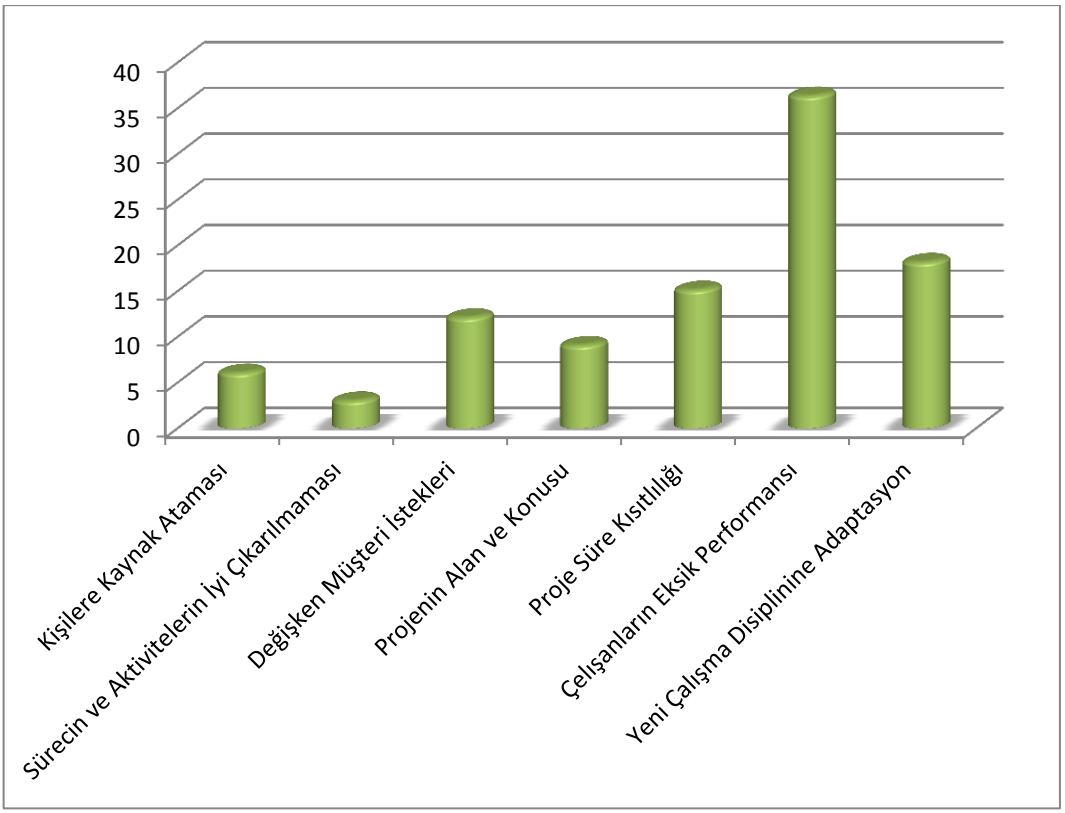
Şekil 6.21 ÇYG dersi için anlaşılabilirlik bakımından en uygun diyagramların yüzdelerini göstermektedir. Sınıf diyagramları kendi başlarına son derece yalın yapılardır [100]. Sınıf diyagramının yalın yapısından dolayı öğrencilerin çoğunluğu anlaşılabilirlik bakımından en uygun diyagramların sınıf diyagram olduğunu düşünmektedirler. Ayrıca use case diyagramının anlaşılabilirlik bakımından yüzdesi göz ardı edilemez.

Öğrencilerin %42'si ise use case diyagramının daha anlaşılır olduğunu düşünmektedirler.



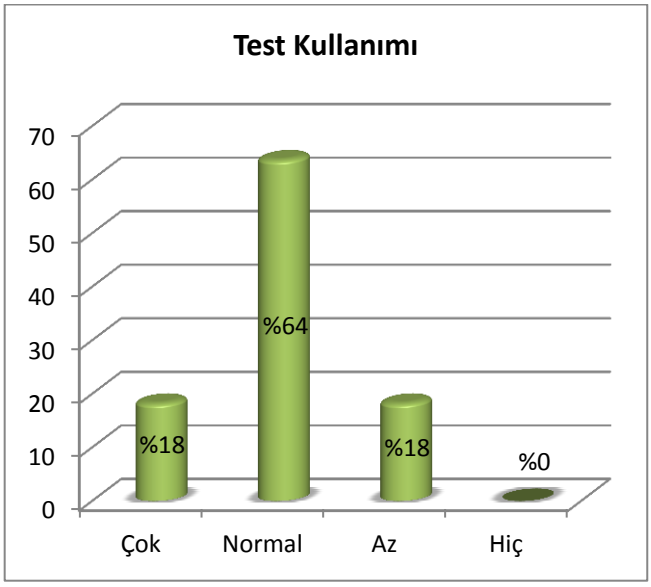
Şekil 6.21 ÇYG dersindeki öğrenciler için anlaşılabilirlik bakımından en uygun diyagram oranları

Öğrencilerin ÇYG ders projesinde, proje tasarım ve analiz sürecinde en çok zorlandıkları alanlar Şekil 6.22'de gösterilmektedir. Öğrenciler en çok ekip içi uyumsuzluk yani ekipteki kişilerin tam olarak çalışmamasından dolayı zorlanırken ikinci olarak yeni çalışma disipline adaptasyonda zorlanmışlardır. Nitekim daha önce de bahsettiğimiz gibi yapılan birçok çalışmada çalışanlar ekip içi uyumsuzluklardan yakınmışlardır ve bu değerlendirmemizde de öğrencilerin en çok bu alanda zorlandıkları sonucunun çıkması doğal bir sonuçtur. İkinci zorlanılan alanın yeni çalışma disipline adaptasyon olması tutarlıdır çünkü ÇYG yeni uygulanmaya başlandığından dolayı öğrenciler bu yeni çalışma disipline adaptasyonda zorlanmışlardır.



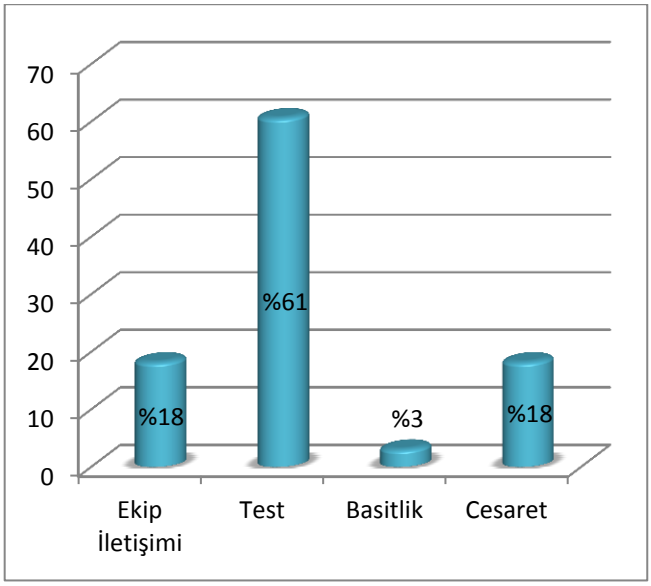
Şekil 6.22 ÇYG ders projesinde, proje tasarım ve analiz sürecinde en çok zorlanılan alanlar ve oranları

Şekil 6.23 ÇYG projesinde öğrencilerin testi ne oranda kullanıldığını göstermektedir. ÇYG projesi test güdümlü olduğundan öğrencilerin yeterince (%64) testi kullanmaları tutarlı bir sonuçtur [101].



Şekil 6.23 ÇYG projesinde test kullanım oranları

Öğrenciler daha önceki projelerin çevik yöntemler ile kıyasladığında öğrencilere göre daha baskın olan farklar Şekil 6.24'te gösterilmektedir. SA ve YM ders projelerinden farklı olarak ÇYG ders projesi test güdümlü bir proje olduğu için "test" öğrenciler için daha baskın bir farktır. Nitekim Şekil 6.23'te de test kullanım oranının yüksek çıkması bu sebeptendir.



Şekil 6.24 Daha önceki projelerin çevik yöntemler ile kıyasladığında öğrencilere göre daha baskın olan farkların oranları

6.2 Veri Madenciliği Tekniklerinden Elde Edilen Sonuçlar

Anketleri sınıflandırmak için NB, BayesNet, MLP, SMO, IBk, KStar ve J48 gibi çeşitli veri madenciliği algoritmaları analiz edildi ve en iyi algoritma seçildi.

En iyi algoritma belirlenirken algoritmaların doğruluk ölçütü oldukça basit ve önemli bir kriterdir ancak sadece doğruluk değerine bakılarak en başarılı algoritma seçilmesi sağlıklı olmaz. Algoritmaların başarıları değerlendirilirken doğruluk değerinin yanı sıra kappa istatistiği, MAE ve RMSE kriterleri de göz önüne alınmıştır. MAE ve RMSE negatif yönelimlidirler ve düşük değerler daha iyidir. Bu yüzden düşük MAE değerine sahip algoritmalar daha iyi algoritmalar olarak değerlendirilir. Bunun değerlendirmelerimiz anket soruları üzerinden yapıldığı için kappa değeri de oldukça önemli bir kriterdir. Kappa değeri ne kadar 1'e yakın olursa doğruluk değerinin soruya verilen cevaptaki yığılmaya bağlı olmadığının yani başarının rastgele olmadığını gösterir.

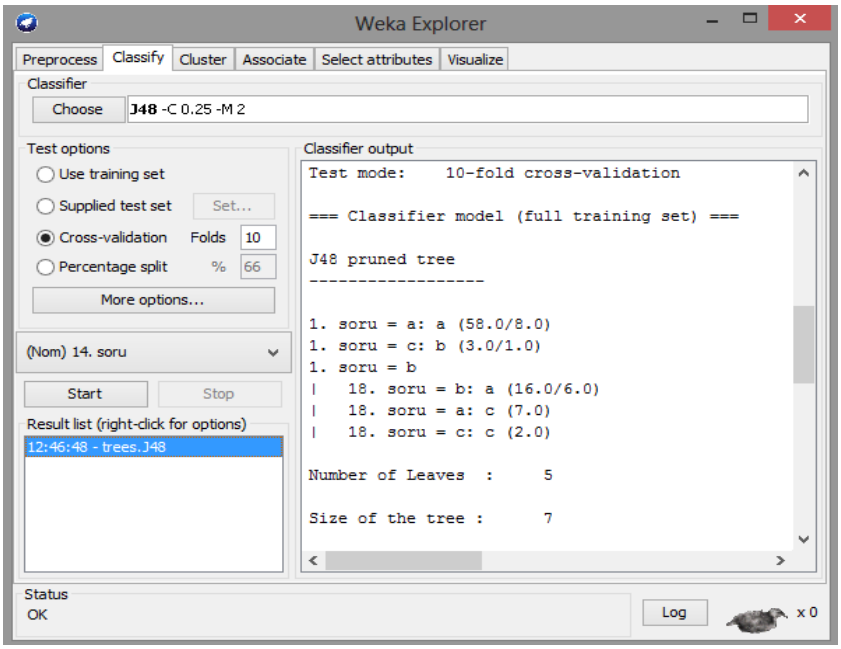
6.2.1 Sistem Analizi ve Tasarımı Ders Proje Anketlerinin Sınıflandırma Sonuçları

SA ders projesinde öğrencilerin projede ER Diyagram, Veri Diyagramı ve Yapı Diyagramından uygulama süresince yararlanma veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçları Çizelge 6.1'de gösterilmektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %80.23, 0.1971, 0.3378, 0.4872 olan J48 algoritmasıdır.

Çizelge 6.1 SA projesinde ER diyagram, veri diyagramı ve yapı diyagramından uygulama süresince yararlanma veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

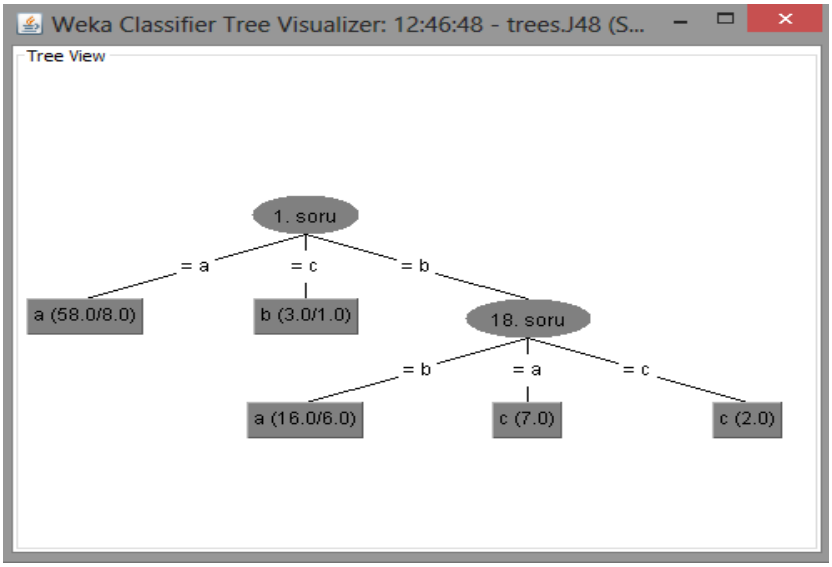
Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	90.70	0.0887	0.2349	0.7771	76.74	0.1845	0.36	0.4195
BayesNet	89.53	0.0853	0.2339	0.7525	74.42	0.1892	0.37	0.4104
MLP	98.84	0.0114	0.0882	0.9732	65.11	0.2315	0.4475	0.2177
SMO	97.67	0.2274	0.2815	0.9471	68.60	0.3101	0.4024	0.2661
IBk	100	0.015	0.0159	1	67.44	0.2209	0.4327	0.1933
KStar	100	0	0	1	68.60	0.2204	0.4199	0.1812
J48	82.56	0.1835	0.3029	0.5199	80.23	0.1971	0.3378	0.4872

J48 algoritmasının çalıştırılması sonucunda elde edilen çıktı Şekil 6.25'te gösterilmiştir. Her satır, ağaçtaki bir düğümü; alt satırlar, ilk satırın çocuk düğümlerini; düğümlerde parantezin içindeki ilk sayı veri kümesindeki kaç vakanın bu düğüm için doğru olarak sınıflandırıldığını; eğer varsa; parantezin içindeki ikinci sayı, düğüm tarafından yanlış olarak sınıflandırılan vakaların sayısını gösterir.



Şekil 6.25 J48 algoritmasının çalıştırılması sonucunda elde edilen çıktı

WEKA'da "classifier output" panelinde metin formatında sunulan karar ağacını "visualize tree" seçeneğini kullanarak görsel olarak ifade etmek mümkündür. Bu ağaç Şekil 6.26'da gösterilmiştir.



Şekil 6.26 J48 karar ağacı

Şekil 6.26'da ilk dallanma öğrencilerin süreç ve raporları ne ölçüde kullandığını; ikinci dallanma sadece veri akış diyagramı verilen bir yazılımın uygulanabilirliği hakkındaki düşüncelerini göstermektedir.

Şekil 6.26'da görülen J48 karar ağacı algoritmasının sonuçlarını aşağıdaki şekilde değerlendirmek mümkündür:

- İlk dallanmada süreç ve rapor kullanımı kısmen olan öğrenci grubunun ayrılma eğilimi taşıdığı görülmektedir.
- Süreç ve raporları büyük ölçüde kullananlar; ER Diyagram, Veri Diyagramı ve Yapı Diyagramından uygulama süresince yararlanmıştır.
- Süreç ve raporları kullanmayanlar; ER Diyagram, Veri Diyagramı ve Yapı Diyagramından uygulama süresince tam istenildiği gibi yararlanmamıştır.
- Süreç ve raporları kısmen kullananlardan, sadece veri akış diyagramı verilen bir yazılımın zor da olsa uygulanabileceğini düşünenler; ER Diyagram, Veri Diyagramı ve Yapı Diyagramından uygulama süresince yararlanmıştır. Sadece veri akış diyagramı verilen bir yazılımın kolaylıkla uygulanacağını veya uygulanamayacağını düşünenler; ER Diyagram, Veri Diyagramı ve Yapı Diyagramından uygulama süresince tam istenildiği gibi yararlanmamıştır.

Çizelge 6.2 veritabanı oluşturulurken ER diyagramından yararlanma veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçlarını göstermektedir. Öğrencilerin ER diyagramından yararlanma sonuçlarına göre sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %68.60, 0.2671, 0.419, 0.4402 olan J48 algoritmasıdır.

Çizelge 6.2 SA projesinde, ER diyagramından yararlanma veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	81.39	0.1664	0.3193	0.6677	62.79	0.2748	0.4367	0.3402
BayesNet	80.23	0.1600	0.3157	0.6488	61.62	0.2785	0.448	0.322
MLP	100	0.0055	0.0082	1	56.98	0.3145	0.5234	0.2262
SMO	95.35	0.2326	0.2905	0.9188	51.16	0.3618	0.4566	0.1652
IBk	100	0.0150	0.0159	1	53.49	0.3234	0.5218	0.1415
KStar	100	0.0000	0.0001	1	55.81	0.3343	0.5011	0.1799
J48	83.72	0.1654	0.2876	0.7118	68.60	0.2671	0.419	0.4402

Çizelge 6.3 süreç ve raporların ne ölçüde kullanıldığı veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçlarını göstermektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %75.58, 0.2407, 0.3615, 0.4414 olan J48 algoritmasıdır.

Çizelge 6.3 SA projesinde süreç ve raporların ne ölçüde kullanıldığı veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	86.05	0.1354	0.2847	0.6775	69.77	0.2309	0.3977	0.2789
BayesNet	88.37	0.1261	0.273	0.7367	72.09	0.2311	0.3992	0.3659
MLP	100	0.0049	0.0082	1	68.60	0.2242	0.4369	0.289
SMO	97.67	0.2274	0.2815	0.9494	70.93	0.3023	0.3916	0.3395
IBk	100	0.015	0.0159	1	60.47	0.2901	0.4758	0.0469
KStar	100	0.0001	0.0002	1	60.47	0.2893	0.4539	0.0571
J48	79.07	0.2178	0.33	0.5212	75.58	0.2407	0.3615	0.4414

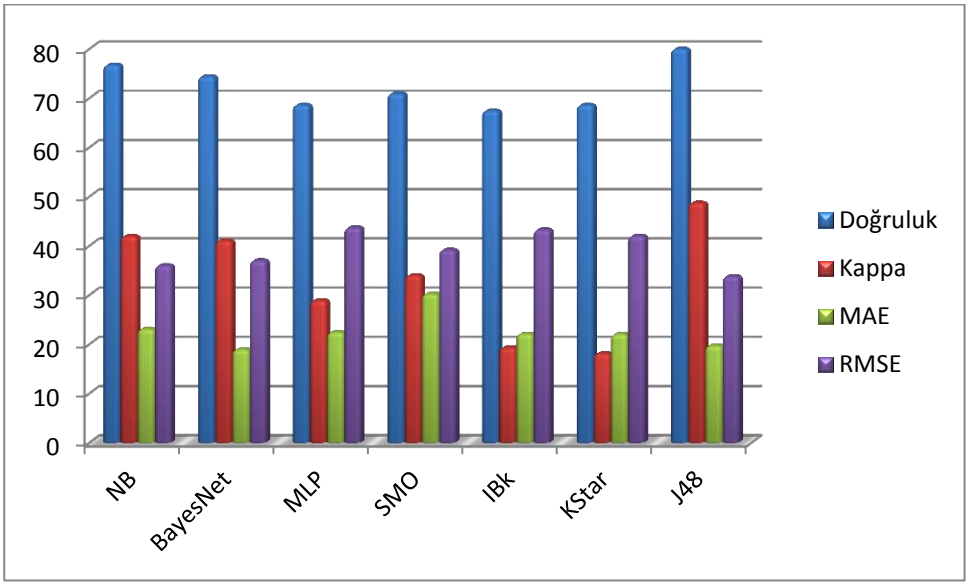
Çizelge 6.4 proje sonunda elde edilen ürünün değerlendirme veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçlarını göstermektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %69.77, 0.2811, 0.4161, 0.3369 olan J48 algoritmasıdır.

Çizelge 6.4 SA dersinde proje sonunda elde edilen ürünün değerlendirme veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	86.05	0.1645	0.275	0.7273	60,47	0.2942	0.4274	0.1878
BayesNet	86.05	0.1535	0.2694	0.7273	59.30	0.2983	0.4397	0.1521
MLP	98.84	0.0129	0.0883	0.9768	58.14	0.2773	0.4958	0.1224
SMO	90.670	0.2429	0.3078	0.8076	61.63	0.3178	0.4109	0.213
IBk	100	0.015	0.0159	1	51.16	0.331	0.5134	-0.0463
KStar	100	0	0.0001	1	52.33	0.33	0.4823	0.0079
J48	77.91	0.2283	0.3378	0.5039	69.77	0.2811	0.4161	0.3369

SA ders projesindeki anket sorularını sınıflandırdığımızda her bir en başarılı algoritmaların karşılaştırma grafikleri Şekil 6.27'de gösterilmiştir. En iyi algoritma

Doğruluk, MAE, RMSE ve KAPPA değerleri göz önüne alındığında sırasıyla %80.23, %19.71, %33.78, %48.72 olan J48 algoritmasıdır.



Şekil 6.27 SA ders proje anketlerinin sınıflandırma algoritmaları sonuçları

6.2.2 Yazılım Mühendisliği Ders Proje Anketlerinin Sınıflandırma Sonuçları

YM ders projesinde öğrencilerin yazılım süreçlerinde ilk oluşturdukları diyagramlar veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçları Çizelge 6.5'te gösterilmektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %60, 0.2436, 0.3998, 0.2629 olan J48 algoritmasıdır.

Çizelge 6.5 YM projesinde öğrencilerin yazılım süreçlerinde ilk oluşturdukları diyagramlar veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	81.43	0.126	0.2438	0.6693	58.57	0.2327	0.3774	0.2448
BayesNet	84.29	0.1142	0.2301	0.7305	57.14	0.2289	0.3816	0.2524
MLP	100	0.0062	0.0103	1	47.14	0.2673	0.4722	0.1178
SMO	98.57	0.2512	0.3128	0.9758	42.85	0.3321	0.4207	0.0123
IBk	100	0.0203	0.0234	1	50	0.2909	0.4891	0.0831
KStar	100	0.0001	0.001	1	47.14	0.2888	0.4685	0.0683
J48	82.86	0.1357	0.2605	0.6908	60	0.2436	0.3998	0.2629

Çizelge 6.6 projelerde modelleme ve tasarımı ifade etmede faydalı diyagram veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçlarını göstermektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %71.43, 0.2485, 0.3769, 0.2036 olan J48 algoritmasıdır.

Çizelge 6.6 YM projesinde modelleme ve tasarımı ifade etmede faydalı diyagram veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	85.71	0.1512	0.28	0.6421	60	0.2787	0.4372	-0,0493
BayesNet	85.71	0.1368	0.2625	0.6429	64.29	0.2779	0.44	0.0458
MLP	98.57	0.0148	0.0979	0.9675	65.71	0.2362	0.4523	0.1558
SMO	98.57	0.2254	0.2779	0.9675	65.71	0.3333	0.4303	0,1541
IBk	100	0.0183	0.0194	1	62.86	0.2696	0.4699	-0.0156
KStar	100	0	0	1	60	0.2721	0.4599	-0.0493
J48	81.42	0.1989	0.3153	0.506	71.43	0.2485	0.3769	0.2036

Çizelge 6.7 müşterinin sürecin ilk başında istediği yazılım ile süreç sonunda oluşturulan yazılımın birbiriyle ne ölçüde örtüştüğü veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçlarını göstermektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %85.71, 0.273, 0.3519, 0.4113 olan SMO algoritmasıdır.

Çizelge 6.7 YM projesinde müşterinin sürecin ilk başında istediği yazılım ile süreç sonunda oluşturulan yazılımın birbiriyle ne ölçüde örtüştüğü veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

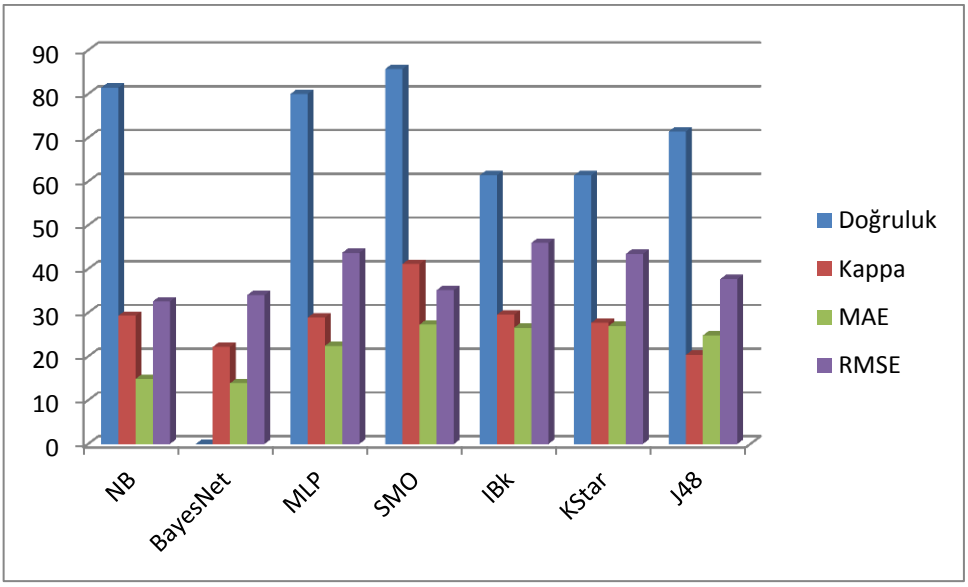
Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	97.14	0.0641	0.1622	0.8961	81.43	0.1484	0.3257	0.2929
BayesNet	95.71	0.054	0.1459	0.8497	77.14	0.1542	0.3396	0.2222
MLP	100	0.0037	0.0066	1	80	0.1389	0.3406	0.216
SMO	100	0.2222	0.2722	1	85.71	0.273	0.3519	0.4113
IBk	100	0.0183	0.0194	1	81.43	0.1486	0.3463	0.036
KStar	100	0	0	1	81.43	0.143	0.3415	0.036
J48	82.86	0.1967	0.3136	0	81.43	0.2033	0.3311	0.0299

Çizelge 6.8 proje gerçekleştirilirken ekip içi uyumunun nasıl olduğu veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçlarını göstermektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %68.57, 0.2302, 0.4479, 0.3949 olan MLP algoritmasıdır.

Çizelge 6.8 YM projesinde ekip içi uyum veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	87.14	0.1493	0.2759	0.7551	65.71	0.2693	0.4075	0.3468
BayesNet	88.57	0.1351	0.2612	0.785	62.86	0.2656	0.4119	0.3019
MLP	100	0.0045	0.0079	1	68.57	0.2302	0.4479	0.3949
SMO	98.57	0.2254	0.2779	0.9729	60	0.3206	0.4153	0.2317
IBk	100	0.0183	0.0194	1	61.43	0.2657	0.4596	0.2956
KStar	100	0	0	1	61.43	0.2696	0.4344	0.2772
J48	77.14	0.2245	0.335	0.5573	57.14	0.314	0.4613	0.1886

YM ders projesindeki anket sorularını sınıflandırdığımızda her bir en başarılı algoritmaların karşılaştırma grafikleri Şekil 6.28'de gösterilmiştir. En iyi algoritma Doğruluk, MAE, RMSE ve KAPPA değerleri göz önüne alındığında sırasıyla %85.71, %27.3, %35.19, %41.13 olan SMO algoritmasıdır.



Şekil 6.28 YM ders proje anketlerinin sınıflandırma algoritmaları sonuçları

6.2.3 Çevik Yazılım Geliştirme Ders Proje Anketlerinin Sınıflandırma Sonuçları

ÇYG ders projesinde öğrencilerin yazılım süreçlerinde ilk oluşturdukları diyagramlar veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçları Çizelge 6.9'da gösterilmektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %75.76, 0.2052, 0.3944, 0.2646 olan J48 algoritmasıdır.

Çizelge 6.9 ÇYG projesinde öğrencilerin yazılım süreçlerinde ilk oluşturdukları diyagramlar veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

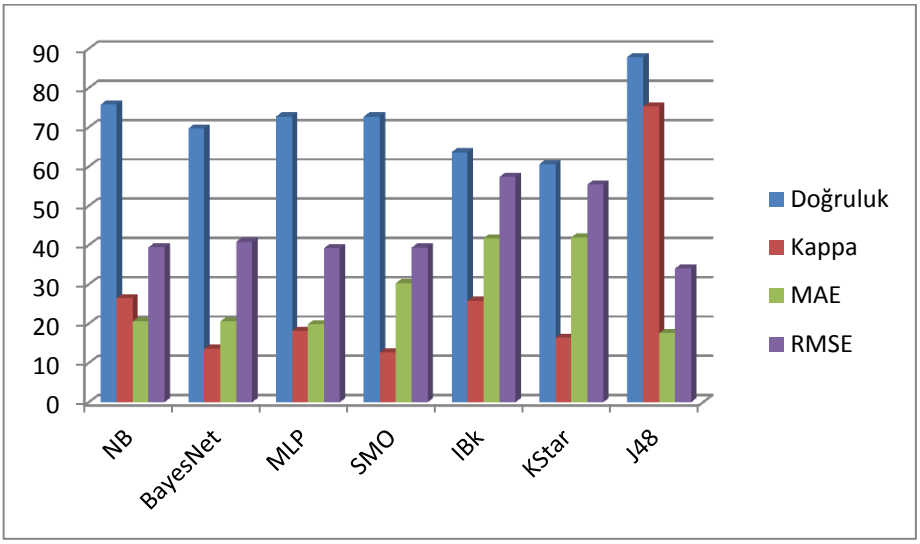
Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	96.97	0.0494	0.1312	0.9255	75.76	0.2052	0.3944	0.2646
BayesNet	93.94	0.0373	0.1265	0.8571	69.7	0.2058	0.4083	0.1361
MLP	100	0.0056	0.0083	1	72.73	0.1989	0.3919	0,1818
SMO	100	0.2222	0.2722	1	72.73	0.303	0.3936	0,1265
IBk	100	0.037	0.0393	1	63.67	0.2572	0.46	0.0109
KStar	100	0	0	1	63.64	0.2453	0.4498	-0.0259
J48	75.76	0.2596	0.3603	0	69.70	0.2772	0.4096	-0.0927

Çizelge 6.10 müşterinin sürecin ilk başında istediği yazılım ile süreç sonunda oluşturulan yazılımın birbiriyle ne ölçüde örtüştüğü veri kümesi için sınıflandırma algoritmalarının Eğitim Fazı ve 10 Katlı Çapraz Onaylama sonuçlarını göstermektedir. Sınıflandırma yapıldığında, en iyi algoritma doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %87.88, 0.1758, 0.3405, 0.7528 olan J48 algoritmasıdır.

Çizelge 6.10 ÇYG projesinde müşterinin sürecin ilk başında istediği yazılım ile süreç sonunda oluşturulan yazılımın birbiriyle ne ölçüde örtüştüğü veri kümesi için eğitim fazı ve 10 katlı çapraz onaylama sonuçları

Algoritma	Eğitim İstatistikleri				10 Katlı Çapraz Onaylama İstatistikleri			
	Doğruluk (%)	MAE	RMSE	Kappa	Doğruluk (%)	MAE	RMSE	Kappa
NB	93.94	0.1234	0.2055	0.8731	57.55	0.4417	0.547	0.061
BayesNet	96.97	0.097	0.1778	0.9374	57.58	0.4434	0.5676	0.061
MLP	100	0.0058	0.0068	1	48.48	0.5296	0.6791	-0.0936
SMO	100	0	0	1	45.45	0.5455	0.7385	-0,1423
IBk	100	0.0286	0.0286	1	63.64	0.4172	0.5736	0.2584
KStar	100	0	0	1	60.61	0.4199	0.554	0.1637
J48	93.94	0.1025	0.2263	0.8731	87.88	0.1758	0.3405	0.7528

ÇYG ders projesindeki anket sorularını sınıflandırdığımızda her bir en başarılı algoritmaların karşılaştırma grafikleri Şekil 6.29'de gösterilmiştir. En iyi algoritma Doğruluk, MAE, RMSE ve KAPPA değerleri göz önüne alındığında sırasıyla %87.88, %17.58, %34.05, %75.28 olan J48 algoritmasıdır.

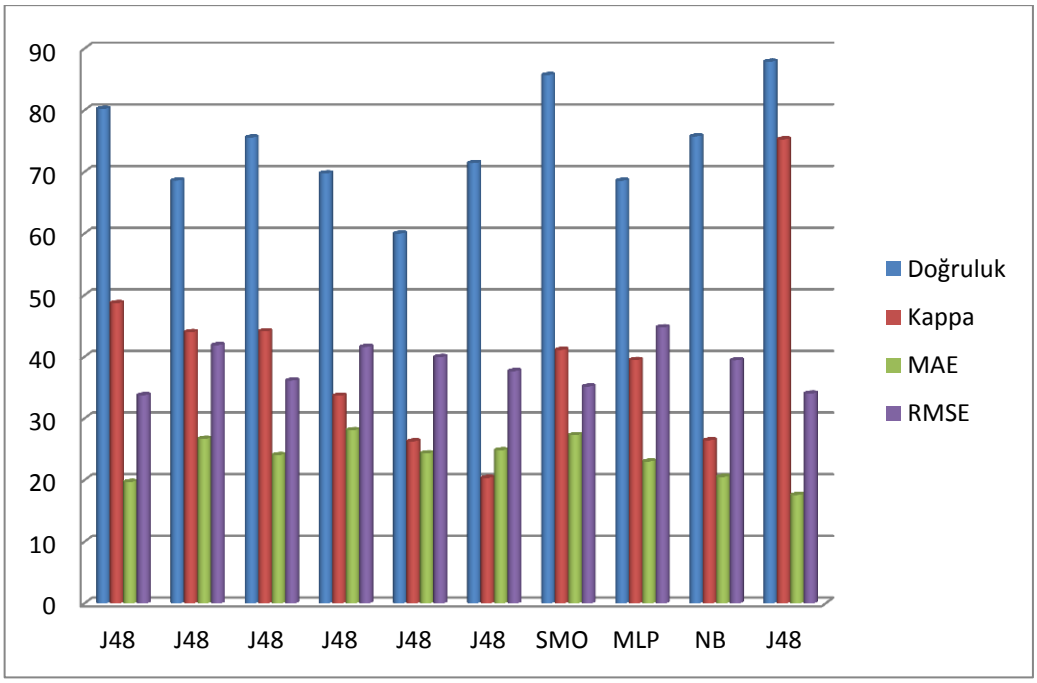


Şekil 6.29 ÇYG ders proje anketlerinin sınıflandırma algoritmaları sonuçları

Veri madenciliğinde bilgiye erişmede farklı metotlar kullanılmaktadır. Bu metotlara ait pek çok algoritma vardır. Bu algoritmalardan hangisinin daha üstün olduğu üzerine pek çok çalışma yapılmış, yapılan bu çalışmalarda farklı sonuçlar elde edilmiştir. Bunun en önemli sebebi, işlem başarımının, kullanılan veri kaynağına, veri üzerinde yapılan önişleme, algoritma parametrelerinin seçimine bağlı olmasıdır. Farklı kişiler tarafından, farklı veri kaynakları üzerinde, farklı parametrelerle yapılan çalışmalarda farklı sonuçlar oluşması doğaldır. Algoritmaların doğruluk ölçütü oldukça basit ve önemli bir kriterdir. Ancak doğruluk ölçütü tek başına yorumlanırsa değerlendirme yanlış sonuçlara götürebilir. Bu ölçütü Kappa, MAE ve RMSE ölçütleriyle beraber ele almak gerekir.

Sınıflandırma için kullanılan algoritmalar ders bazında değerlendirildiğinde SA ve ÇYG ders proje anketlerinin sınıflandırılması için en başarılı algoritma J48 iken YM ders proje anketlerinin sınıflandırılması için en başarılı algoritma SMO algoritmasıdır.

Tüm ders proje anketlerini tek bir algoritma ile sınıflandırıp en başarılı algoritmayı belirlemek istediğimizde, toplamda 10 soru üzerinde 7 farklı sınıflandırma algoritmasının soru bazında başarıları Doğruluk, Kappa, MAE ve RMSE sonuçları Şekil 6.30'da gösterilmektedir.



Şekil 6.30 Her sorunun değerlendirilmesinde elde edilen en başarılı algoritma ve bu algoritmanın kappa, MAE ve RMSE sonuçları

Soru bazında algoritmaların sınıflandırma sonuçları incelendiğinde yedi soruda J48 algoritması, bir soruda SMO, bir soruda MLP algoritması ve bir soruda Naive Bayes algoritması başarılı çıkmıştır. Bu da soruların büyük çoğunluğunda J48 algoritmasının başarılı olduğunu gösterir. Tüm sorular üzerinde uygulanan sınıflandırma algoritmaları incelendiğinde doğruluk, MAE, RMSE ve KAPPA değerleri sırasıyla %87.88, 0.1758, 0.3405, 0.7528 olan J48 algoritması en iyi sonucu üretmiştir. Nitekim çalışmalarda da, bir karar ağacı algoritması olan C4.5 algoritmasının diğer algoritmalardan daha iyi sonuç ürettiği sonucuna ulaşıldığı belirtilmiştir [62], [102], [103], [104].

6.2.4 Birliktelik Kuralı Sonuçları

Şekil 6.31 Yazılım Mühendisliği anket verilerine Weka'da Apriori algoritması uygulandıktan sonra elde edilen kuralları göstermektedir. Kurallardan bazıları incelendiğinde, gelecekle ilgili tahminlerin de yapılabileceği sonucu ortaya çıkmaktadır. %97 güvenilirlikle elde ettiğimiz kurala göre (1. kural);

- Yazılım süreçlerinde ilk kullanıcı senaryosu diyagramlarını oluşturan,
- Projelerinde karmaşık sınıfların sayısı az olan,

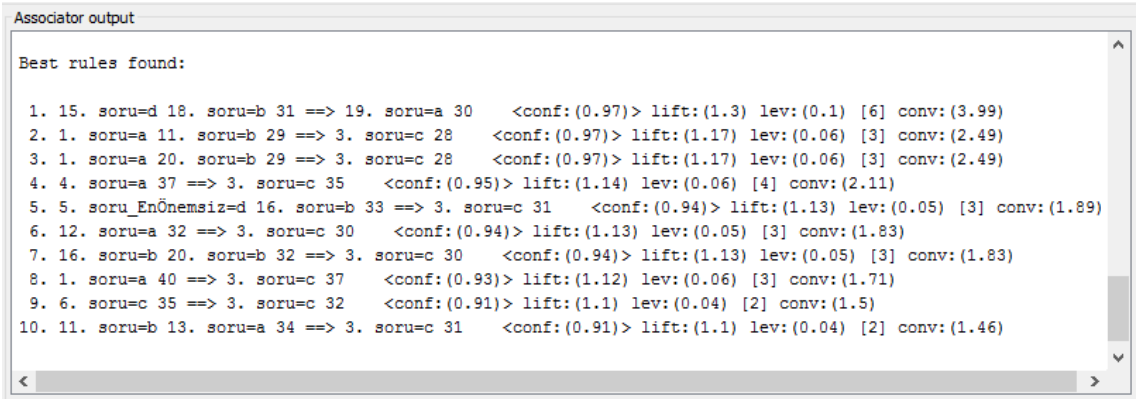
- Öğrencilerin projelerinde, nesneye dayalı tasarımda anlaşılabilirlik bakımından en uygun diyagram onlar için sınıf diyagramı olacaktır.

Sınıf diyagramları en temel diyagramlar olup son derece yalın yapıda olduğundan anlaşılması oldukça kolaydır [98], [99]. Bu nedenle öğrencilerin projelerde daha karmaşık diyagramları kullanmasındansa, sınıf diyagramını uygun görüp kullanımını arttırmak için projelerde karmaşık sınıfların sayısını azaltmak gerekir. Çünkü bir yazılım mühendisinin yaptığı ilk şey kullanıcı senaryosu diyagramlarını çizmek [95] olduğundan, sınıf diyagramın en uygun diyagram görülmesi karmaşık sınıfların sayısına bağlı olmaktadır.

Ayrıca yine %97 güvenilirlikle elde ettiğimiz diğer bir kurala göre (2. kural);

- Süreç ve raporlamayı büyük ölçüde kullanan,
- Fizibilite analizi yapmak için MS Project kullanan,
- Öğrencilerin projeleri, süreç başında ile süreç sonunda büyük ölçüde örtüşecektir.

Dokümantasyon son derece önemli olup, iyi yapılmaması durumunda yazılım geliştirmede verimlilik ve kalite bakımından önemli bir eksikliğe yol açabilir. Bu nedenle de öğrenciler derste önerilen yöntemi kullanıp dokümantasyonu büyük ölçüde yaptıklarında, projenin kalitesi ve verimi artacak bu da süreç başında istenilen ürüne paralel bir ürün elde etme olasılığını arttıracaktır.

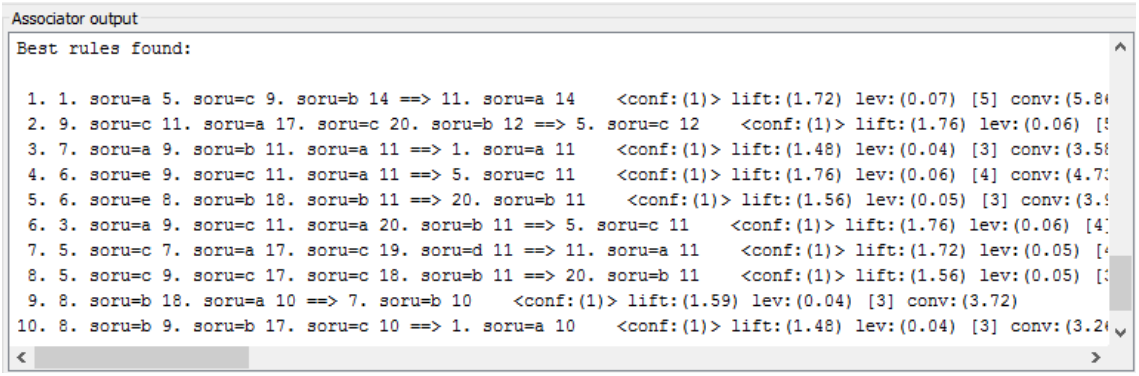


Şekil 6.31 YM anket verilerine apriori uygulandıktan sonra elde edilen kurallar

Şekil 6.32 SA anket verilerine Apriori algoritması uygulandıktan sonra elde edilen kuralları göstermektedir. %100 güvenirlikle elde ettiğimiz kurala göre (1.kural);

- Ekip içi uyumları iyi olan,
- Süreç ve raporlamayı büyük ölçüde kullanan,
- Sistem analiz ve tasarım aşamaları tüm yazılım geliştirme sürecinin bir kısmını oluşturan,
- Öğrencilerin projelerinde, MS ile belirlenen yazılım geliştirme süresi ile proje geliştirme süresi birbirine paralellik gösterecek ve zaman aşılmayacaktır.

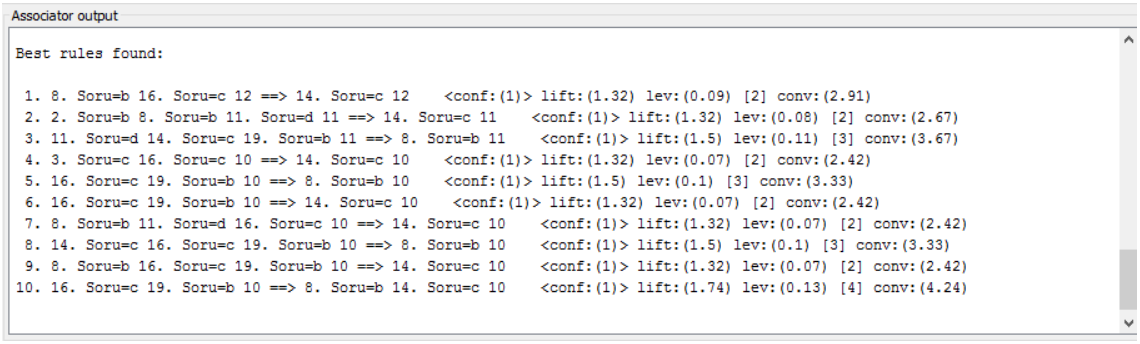
Anket sorularından olan proje tasarım ve analiz sürecinde en çok hangi alanda zorlandınız sorusuna verilen cevaplar incelendiğinde öğrencilerin en çok zorlandıkları alanlardan bir tanesi proje süre kısıtlılığıdır. SA dersi sonunda projelerde süre sıkıntısı yaşanmaması için elde edilen kural oldukça önemlidir. Yani sürenin aşılmamışı için ekip içi uyumun iyi olup dokümantasyonun büyük ölçüde yapılması gerekmektedir.



Şekil 6.32 SA anket verilerine apriori uygulandıktan sonra elde edilen kurallar

Şekil 6.33 ÇYG anket verilerine Weka'da Apriori algoritması uygulandıktan sonra elde edilen kuralları göstermektedir. %100 güvenilirlikle elde ettiğimiz kurala göre (2.kural);

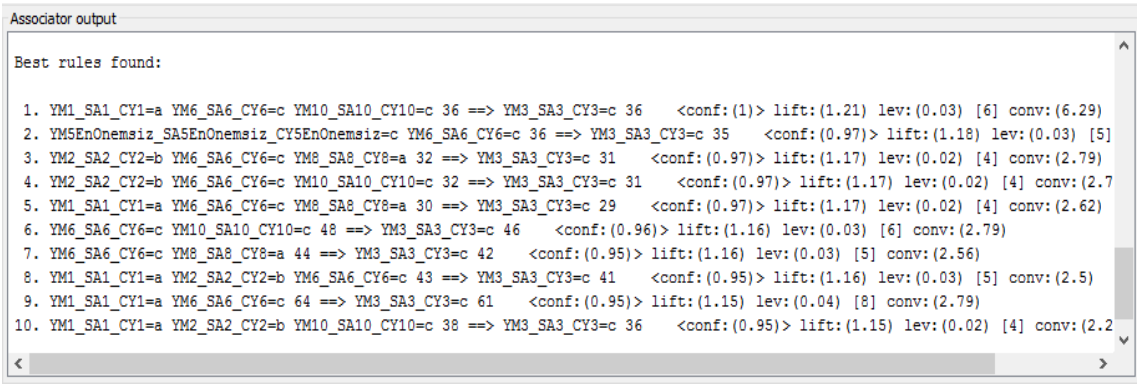
- Süreç ve raporlamayı büyük ölçüde kullanan,
- Projesinin geliştirilmesi gerektiğini düşünen,
- ÇYG ders projesini diğer derslerdeki projelerle kıyasladığında *geliştirilen yazılımın testleri* konusunda daha fazla katkı sağlamıştır diyen,
- Öğrenciler projelerinde, ilk kullanıcı senaryoları diyagramlarını oluşturacaktır.



Şekil 6.33 ÇYG anket verilerine apriori uygulandıktan sonra elde edilen kurallar

Ortak anket sorularından elde ettiğimiz veri setine Weka'da Apriori algoritması uygulandıktan sonra elde edilen sonuçları Şekil 6.34'de göstermektedir. %100 güvenirlikle elde ettiğimiz kurala göre; süreç ve raporlamayı büyük ölçüde kullananların ekip içi uyumu orta olan grupların projeleri tüm yazılımın %50-75'ini oluşturuyorsa, müşterinin süreç başında ile sonunda istediği yazılım birbiriyle büyük ölçüde örtüşecektir. Güvenirliği %97 olan başka bir kurala göre (5.kural);

- Süreç ve raporlamayı büyük ölçüde kullanan,
- Ekip içi uyumları iyi olan,
- Proje sonunda elde edilen ürünü iyi olarak değerlendiren,
- Öğrencilerin projeleri, sürecin başında ile sonunda istenilen yazılım birbiriyle büyük ölçüde örtüşecektir.



Şekil 6.34 Ortak anket sorularından elde edilen kurallar

BÖLÜM 7

SONUÇ VE ÖNERİLER

İki farklı üniversitenin bilgisayar mühendisliği bölümündeki yazılım geliştirme derslerinden olan *Sistem Analizi ve Tasarımı*, *Yazılım Mühendisliği* ve *Çevik Yazılım Geliştirme* derslerinde toplamda 189 öğrenciye ders projeleri ile ilgili anketler yapılmıştır. Derslerde gerçekleştirilen projelerin iyileştirilmesi amacıyla öğrencilerin sosyal ve teknik yeteneklerini ölçecek şekilde hazırlanan anket soruları analiz edilmiştir. Teknik yetenek soruları incelendiğinde *Sistem Analizi ve Tasarımı* dersinde öğrencilerin sadece %6'sı projesinde kullanılması gereken diyagramları kullanmamış ve öğrencilerin %10'u veritabanı oluşturulurken yararlanılması gerekilen veri akış diyagramından yararlanmamıştır. *Yazılım Mühendisliği* dersinde öğrencilerin %56'sı projesinin yazılım sürecinde ilk oluşturduğu diyagram kullanıcı senaryosu diyagramıdır ve öğrencilerin %69'u ardışıl ve sınıf diyagramlarını tasarım aşamasında kullanmıştır. *Çevik Yazılım Geliştirme* dersinde ise öğrencilerin %76'sı projesinin yazılım sürecinde ilk oluşturduğu diyagram kullanıcı senaryosu diyagramıdır ve %85'i kullanıcı senaryosu ve sınıf diyagramlarını, analiz ve tasarım aşamasında kullanmıştır. Elde edilen bu sonuçlar bize öğrencilerin büyük bir kısmının, projelerini derste anlatılan bilgiler çerçevesinde gerçekleştirdiğini göstermektedir.

Sosyal yetenek soruları analiz edildiğinde öğrencilerin sadece %55'i dokümantasyonu büyük ölçüde kullanmaktadır. Dokümantasyon yazılım geliştirmede verimlilik ve kalite bakımından önemli bir eksikliğe yol açabilir. Bu nedenle öğrencilerin dokümantasyonu büyük ölçüde yapma oranı oldukça düşüktür. Öğrencilerin %47'si proje içi ekip uyumunu iyi bulmamaktadır. Öğrenciler grupları oluştururken, grup arkadaşlarını kendileri seçtikleri halde yarısından fazlası ekip uyumunu beğenmemektedir. Ekip içi

uyum sektörde önemli bir yerde olup başarısızlıkların nedenlerinin başında gelmektedir. Bu nedenle ekip uyumu sektörde önemli bir yerdedir. Öğrencilerin %49'u projedeki süre kısıtlılığı ve çalışanların çalışmamalarından dolayı zorlanmaktadırlar. Öğrencilerin projelerini daha verimli gerçekleştirmeleri için projeler bir dönemde değil daha uzun bir sürece yayılırsa başarı artacaktır. Ayrıca öğrencilerin sadece %31'i süreç sonunda elde edilen ürünü iyi bulmaktadır.

Elde edilen teknik yetenek sonuçları ile sosyal yetenek sonuçları karşılaştırıldığında, öğrencilerin sosyal yeteneklerinin teknik yetenekler kadar iyi olmadığı sonucu elde edilmiştir. Yapılan çalışmalarda sosyal yeteneklerin, teknik yetenekler kadar önemli olduğunu ve sosyal yeteneklerin başarısız olması durumunda tüm projenin başarısız olduğu ifade edilmiştir. Elde edilen bu sonuçlar doğrultusunda akademinin sosyal yeteneklere önem verip bu yetenekleri iyileştirilmesi durumunda daha kaliteli yazılım mühendisliği mezunları yetiştirilebilir böylelikle endüstrilerde başarılı projelerin oranı artacaktır. Dolayısıyla endüstrinin beklentilerini karşılayan öğrenciler yetişmiş olacaktır. Böylelikle endüstrinin, kalitesiz yazılım mühendisliği mezunları yetiştiriliyor iddiası azalacak ve akademi ile sektörler arasındaki boşluk azalacaktır. Bu sayede hem akademi hem de endüstri birçok fayda görecektir.

Tez kapsamında ders projelerini sınıflandırmak için çeşitli veri madenciliği algoritmalarından olan NB, BayesNet, MLP, SMO, IBk, KStar ve J48 algoritmaları analiz edilip en iyi algoritma seçilmiştir. En iyi algoritma belirlenirken doğruluk değeri, kappa istatistiği, MAE ve RMSE kriterleri göz önüne alınmıştır. Bu değerlendirmeler sonucunda; *Sistem Analizi ve Tasarımı* ders proje anketlerinden çeşitli sorular sınıflandırıldığında en iyi algoritma J48 algoritması, *Yazılım Mühendisliği* ders proje anketlerinden çeşitli sorular sınıflandırıldığında en iyi algoritma SMO algoritması, *Çevik Yazılım Geliştirme* ders proje anketlerinden çeşitli sorular sınıflandırıldığında en iyi algoritma J48 algoritması olmuştur. Tüm dersler için ortak bir sınıflandırma algoritması seçilecek olursa en iyi başarıya sahip olan algoritma C4.5 algoritmasının Weka implementasyonu olan J48 karar ağacı algoritması daha başarılı bulunmuştur. Nitekim çalışmalarda da, bir karar ağacı algoritması olan C4.5 algoritmasının diğer algoritmalardan daha iyi sonuç ürettiği sonucuna ulaşıldığı belirtilmiştir [102], [103], [104].

Ayrıca tez kapsamında anket soruları arasındaki beklenmeyen ilişkileri bulmak, gizli bilgileri açığa çıkararak gelecekteki eğilimleri belirlemek için, veri madenciliğinde, birliktelik kuralı çıkarım algoritmalarından biri olan Apriori algoritması kullanılmıştır. Elde edilen kurallara göre; dokümantasyonu iyi yapan öğrenciler, fizibilite analizi yapmak için MS Project kullanmışlar ise projeleri süreç başında ile süreç sonunda büyük ölçüde örtüşecektir. Diğer bir kurala göre, ekip içi uyumun iyi olduğu gruplar projelerinde dokümantasyonu iyi yapmış ve projelerinde sistem analiz ve tasarım aşamaları tüm yazılım geliştirme sürecinin bir kısmını oluşturuyorsa, geliştirilen projeler MS ile belirlenen yazılım geliştirme süresi ile proje geliştirme süresi birbirine paralellik gösterecek ve zaman aşılmayacaktır. Başka bir kurala göre, süreç ve raporlamayı büyük ölçüde kullanan öğrencilerin ekip içi uyumu iyi ve proje sonunda elde edilen ürünü iyi olarak değerlendiriyorlarsa, sürecin başında ile sonunda istenilen yazılım birbiriyle büyük ölçüde örtüşecektir.

Sonuç olarak tez kapsamında elde ettiğimiz sonuçlara göre projeler iyileştirilir, ders kapsamları elde edilen bilgiler çerçevesinde zenginleştirilirse derslerde yapılan projeler daha başarılı olacaktır.

İleride yapılacak çalışmada daha iyi sınıflandırma başarısı alabilmek için kullanılan verilerden daha fazla sayıda veri kullanılmalıdır. Öğrenciler hakkında daha fazla çıkarım yapabilmek için soru çeşitliliği attırılabilir. Projelerdeki durumlarını daha iyi anlayabilmek için sorular arasındaki daha fazla ilişki kurulabilir. Projelerdeki zaman sorununu ortadan kaldırmak için daha uzun bir süreçte projeler gerçekleştirilebilir. Ayrıca projeler endüstrilerle gerçekleştirilirse hem öğrenciler deneyim kazanmış olur hem de endüstride gerçekleştirilen projelerin taklidi olarak değil gerçek dünyadan projeler ile gerçekleştirmiş olacaklardır.

KAYNAKLAR

[1] Software Engineering 2004, Software Engineering Volume, http://sites.computer.org/ccse/, Kasım 2012.

[2] Cifuentes, C. ve Hughes, J., (1994). "SE curriculum design: methodologies, formal methods, and life cycle models.", II. Formal methods Software Education Conference.

[3] Pullan, W. ve Oliver, D., (1994). "Development of an undergraduate software engineering degree", Software Education Conference, 111-117.

[4] Bagert, D.J, (1998). "The challenge of curriculum modeling for an emerging discipline: software engineering", 910-915.

[5] Schneider, J.-G., Johnston, L. ve Joyce, P., (2005). "Curriculum development in educating undergraduate software engineers - are students being prepared for the profession?", Software Engineering Conference, Australian, 314-323.

[6] Bourque, P. ve Dupuis, R., (2004). Guide to the Software Engineering Body of Knowledge, IEEE, Los Alamitos, California.

[7] C. Liu, (2005). "Enriching Software Engineering Courses with Service- Learning Projects and the Open-Source Approach", ICSE'05, St. Louis, Missouri, USA, 613-614.

[8] Bagert, D.J., Hilburn, T.B., Hislop, G.W. ve Mengel, S.A., (1998). "Guidelines for software education: meeting the needs of the 21st Century", Frontiers in Education Conference.

[9] Mohay, G., Morarji, H. ve Thomas, R., (1994). "Undergraduate, graduate and professional education in software engineering in the '90's: a case study", Software Education Conference, 103-110.

[10] Ghezzi, C. ve Mandrioli, D., (2005). "The challenges of software engineering education", Software Engineering, 27th International Conference, 637-638.

[11] Morrogh, P., (2000). "Is software education narrow-minded? - A position paper", Software Engineering, 2000. Proceedings of the 2000 International Conference, 545-546.

[12] Saiedian, H., Bagert, D. ve Mead,N., (2002). "Software Engineering Programs: Dispelling the Myths and Misconceptions", IEEE Software, 19(5):35-41.

[13] Hilburn, T. ve Humphrey, W., (2002). “The Impending Changes in Software Education”, IEEE Software, 19(5):22-24.

[14] Hawthorne, M. J. ve Perry, D. E., (2005) “Software Engineering Education in the Era of Outsourcing, Distributed Development, and Open Source Software: Challenges and Opportunities”, St. Louis, Missouri, USA, 643-644.

[15] Ciancarini, P., (2005). “On the Education of Future Software Engineers”, ICSE'05, St. Louis, Missouri, USA, 649-650.

[16] Hazzan, O. ve Tomayko, J., (2005). “Teaching Human Aspects of Software Engineering”, ICSE'05, St. Louis, Missouri, USA, 647-648.

[17] Hilburn,T.B. ve Humphrey, W.S., (2002). “The Impending Changes in Software Education”, IEEE Software, 19(5):22-25.

[18] Tamai,T., (2005). “How to Teach Software Modeling”, ICSE'05, St. Louis, Missouri, USA, 609-610.

[19] Kruchten, P., (2004). “Putting the 'Engineering' into Software Engineering,”Australian Software Eng. Conf. (ASWEC 2004), Melbourne, Australia, 2-8.

[20] Parnas, D., (1999). “Software Engineering Programs Are Not Computer Science Programs” IEEE Software, 16(6): 19-30.

[21] Pour, G., Griss, M. ve Lutz, M., (2000). "The Push to Make Software Engineering Respectable", IEEE Computer, 33(5): 35-43.

[22] Abran, A. ve Moore, J., (2001), “Guide to the Software Engineering Body of Knowledge SWEBOK”. IEEE Computer Society Press, 2001.

[23] Software Engineering 2004, Computing Curricula, http://www.computer.org/education/cc2001/, Aralık 2012.

[24] The CHAOS report, http://www.csus.edu/indiv/v/velianitis/161/ChaosReport.pdf, Aralık 2012.

[25] Chaos, A., (1999) “Recipe for success, The Standish Group Int’1”, West Yarmouth, Mass., 1999.

[26] Project Management, Standish 2009 CHAOS Report, http://itprojectguide.blogspot.com/2009/04/standish-2009-chaos-report.html, Aralık 2013.

[27] Hislop G.W., (2009). “Software Engineering Education: Past, Present and Future”, Software Engineering Effective Teaching and Learning Approaches and Practices, Information Science, 2009.

[28] van Vilet, H., (2006). “Reflections on software engineering education”, Software, IEEE, 23(3): 55-61.

[29] Oudshoorn, M.J. ve Maciunas, K.J., (1994). “Experience with a project-based approach to teaching software engineering”, Software Education Conference, 220-225.

[30] Morgan, G.W. ve Lear, F.A., (1994). “The role of a software engineering project within an undergraduate applied computing degree”, Software Education Conference, 230-236.

[31] Aizamil, Z., (2005). “Towards an Effective Software Engineering Course Project”, ICSE'05, 15-21 May, St. Louis, Missouri, USA, 631-632.

[32] Dawson, R. ve Newsham, R., (1997). “Introducing Software Engineers to the Real World” IEEE Software, 14(6):37-43.

[33] Dawson, R., (2000). “Twenty Dirty Tricks to Train Software Engineers” 22nd Int'l Conf. Software Eng. (ICSE 00), 209-218.

[34] Yamaura, T. ve Onoma, A.K., (2002). “University software education matched to social requests”, Cyber Worlds Proceedings First International Symposium, 331-336.

[35] Ellis, H.J.C., Mead, N.R., Moreno, A.M. ve Seidman, S.B., (2003). “Industry/University software engineering collaborations for the successful reeducation of non-software professionals”, Software Engineering Education and Training (CSEE&T 2003), 44-51.

[36] Vliet, H., (2005). “Some Myths of Software Engineering Education”, ICSE'05, 15-21 May 2005, St. Louis, Missouri, USA, 621-622.

[37] Albayrak, O., (2003). “Bilgisayar Mühendisliği Eğitimine Katkısı Olacak Öneriler”, I. Elektrik Elektronik Bilgisayar Mühendislikleri Eğitimi Sempozyumu, Ankara, 2003.

[38] Shami, N. S., (2006). “Together apart: An ethnographic study of industry academia collaboration”, Proceedings of Supporting the Social Side of Large Scale Software Development a CSCW Workshop, Banff, AB.

[39] Begel, A. ve Simon, B., (2008). “Novice Software Developers, All Over Again”, ICER’08, Sydney, Australia.

[40] Begel, A. ve Simon, B., (2008). “Struggles of New College Graduates in their First Software Development Job”, SIGCSE’08, Portland, Oregon, USA.

[41] Denning, P.J., (1992). “Educating a New Engineer”, Communications of the ACM, 35(12):82-97.

[42] Almi, N.E.A.M., Rahman, N.A., Purusothaman, D. ve Sulaiman, S., (2011). “Software Engineering Education: The Gap Between Industry’s Requirements and Graduates’ Readiness”, IEEE Symposium on Computers & Informatics (ISCI), Kuala Lumpur, 542- 547.

[43] Rusu, A., Rusu, A., Docimo, R., Santiago, C. ve Paglione, M., (2009). “Academia-Academia-Industry Collaborations on Software Engineering Projects Using Local-Remote Teams”, SIGCSE’09, Chattanooga, Tennessee, USA.

[44] Beckman, H., Coulter, N., Khajenoori, S. ve Mead, N.R., (1997). “Collabaration closing the industry academia gap”, Software IEEE, 14(6):49-57.

[45] Aybay, I., (2005). “Yazılım Mühendisliği Dersi için Proje Ağırlıklı ve Problem Çözmeye Dayanan Yeni Bir Yaklaşım”, II. Ulusal Yazılım Mühendisliği Sempozyumu, Ankara.

[46] Teles, V. M. ve de Oliveira, C.E.T., (2003). “Reviewing the Curriculum of Software Engineering Undergraduate Courses to Incorporate Communication and interpersonal skills teaching”, Software Engineering Education and Training (CSEE&T 2003), 158-165.

[47] Kumar, V., Kinzel, G., Wei, S., Bengu, G., ve Zhou, J., (2000). “Multi-university design projects”, Journal of Engineering Education, 89(3):353.

[48] Liu, C., (2005). “Enriching software engineering courses with service-learning projects and the open-source approach”, In Proceedings of the 27th international Conference on Software Engineering, St. Louis, MO, USA, 613-614.

[49] Reichlmayr, T.J., (2006). “Collaborating With Industry–Strategies for an Undergraduate Software Engineering”, Proceedings of the 2006 international workshop on Summit on software engineering education, Shanghai, China, 13-16.

[50] Rusu, A., Rusu, A., Paglione, M., Snyder, F., Santiago, C., (2007). “Overcoming Limited Resources: An Academia-Government Partnership on Software Engineering and Capstone Projects”, Proceedings of the 37th ASEE/IEEE Frontiers in Education Conference.

[51] 9ncu Kalkınma Planı, www.kalkinma.gov.tr/DocObjects/View/13744/plan9.pdf, DPT, Ankara.

[52] Beckman, K., Coulter N., Khajenoori, S. ve Mead, N.R., (1997). “Collaboration: Closing the Industry-Academia Gap”, IEEE Software, 14(6):40-57.

[53] Kajko-Mattsson, M., (2012). “A Method for Designing Software Engineering Educational Programs”, 25th IEEE Conference on Software Engineering Education and Training, Klagenfurt, Austria, 139-143.

[54] Ardis, M.A., Chenoweth, S.V. ve Young, F.H., (2008). “The “Soft” topics in software engineering education”, Frontiers in Educaton Conference, Saratoga Springs, New York, F3H-1-F3H-6.

[55] Bollin, A., Hochmuller, E., Mittermeir, R. ve Samuelis, L., (2012). “Experiences with Integrating Simulation into a Software Engineering Curriculum”, 25th Conference on Software Engineering Education and Training (CSEE&T), Nanjing, Jiangsu, 62-71.

[56] Albayrak, Ö., (2007). “Yazılım Mühendisliği Eğitimi: Gereksinim Belirleme ve Analiz Aşamasına Yönelik Deneyim ve Uygulamalar”, III. Ulusal Yazılım Mühendisliği Sempozyumu ve Sergisi Bildiriler Kitabı, Ankara, 15-18.

[57] Blake, M.B., (2003). “A Student-Enacted Simulation Approachto Software Engineering Education”, IEEE Transactıons on Educatıon, 46(1):124-132.

[58] Walker, E. L. ve Slotterbeck, O.A., (2002). “incorporating realistic teamwork into a small college software engineering curriculum”, Journal of Computing Sciences in Colleges, 17(6):115-123.

[59] Souza C. R. B., (2006). “Exploring the Relationship between Software Dependencies and the Coordination of Software Development Work”, Proceedings of Supporting the Social Side of Large Scale Software Development a CSCW Workshop, Banff, AB.

[60] Trainer, E., Quirk, S., Cleidson de Souza, ve Redmiles, D., (2005). “Bridging the Gap between Technical and Social Dependencies with Ariadne” Proceedings of OOPSLA Workshop on Eclipse Technology Exchange, San Diego, CA, 26-30.

[61] Sillito, J. ve Wynn, E., (2006). “Social Dependencies and Contrasts in Software Engineering Practice”, Proceedings of Supporting the Social Side of Large Scale Software Development a CSCW Workshop, Banff, AB.

[62] Bhullar, M.S., IAENG, M. ve Kaur, A., (2012). “Use of Data Mining in Education Sector”, Proceedings of the World Congress on Engineering and Computer Science 2012 Vol I WCECS 2012, San Francisco, USA.

[63] Agarwal S., Pandey, G. N. ve Tiwari, M. D., (2012). “Data Mining in Education: Data Classification and Decision Tree Approach”, International Journal of e-Education, e-Business, e-Management and e-Learning, 2(2):140-144.

[64] Bunkar, K., Singh, U.K., Pandya, B. ve Bunkar, R., (2011). “Data Mining: A prediction of performer or underperformer using classification”, Ninth International Conference on Wireless and Optical Communications Networks (WOCN), Indore, India, 1-5.

[65] Sun, H., (2010). “Research on Student Learning Result System based on Data Mining”, IJCSNS International Journal of Computer Science and Network Security, 10(4):203-205.

[66] Bozkır, A.S., Sezer, E. ve Gök, B., (2009). “Öğrenci Seçme Sınavında (ÖSS) Öğrenci Başarımını Etkileyen Faktörlerin Veri Madenciliği Yöntemleriyle Tespiti”, 5. Uluslararası İleri Teknolojiler Sempozyumu (IATS’09), Karabük Üniversitesi, Karabük, 37-43.

[67] Silahtaroğlu, G., (2008). Kavram ve Algoritmalarıyla Temel Veri Madenciliği, Papatya Yayıncılık, İstanbul, Türkiye.

[68] Terzi, Ö., Küçüksille, E. U., Ergin G. ve İlker, A., (2011). “Veri Madenciliği Süreci Kullanılarak Güneş Işınımı Tahmini”, SDU International Technologic Science, 3(2):29-37.

[69] Wei C., Chiu T., (2002). “Turning telecommunications call details to churn prediction : a data mining approach”, Expert Systems with Applications, 23(2):103-102.

[70] Quinlan, J.R., (1986). “Induction of Decision Trees”, Machine Learning, 1(1):81-106.

[71] Kaariainen, M. ve Malinen, T. (2004). “ Selective Rademacher Penalization and Reduced Error Pruning of Decision Trees”, Journal of Machine Learning, 5: 1107-1126.

[72] Tan, P.N., Steinbach, M. ve Kumar, V., (2006). Introduction to Data Mining, USA.

[73] Data Mining Software in Java, www.cs.waikato.ac.nz/~ml/weka/, Mart 2013.

[74] Ardıl, E., (2009). Esnek Hesaplama Yaklaşımı İle Yazılım Hata Kestirimi, Yüksek Lisans Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Edirne.

[75] Pelikan M., Goldberg, D. E. ve Erick, C.P., (1999). "BOA: The Bayesian Optimization Algorithm", Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99, Orlando, Florida, USA, 525-532.

[76] Aha, D. ve Kibler, D. (1991). "Instance-based learning algorithms", Machine Learning, 6(1):37-66.

[77] Cleary, J.G. ve Trigg, L.E. (1995). "K*: An Instance-based Learner Using an Entropic Distance Measure", Proceedings Tweleveth International Conference on Machine Learning, Tahoe City, California, 108-114.

[78] Agrawal, R. ve Srikant, R., (1994). "Fast Algorithms for Mining Association Rules", Proceedings of the 20th International Conference on Very Large Databases (VLDB '94), 487-489.

[79] Zhu H., (1998). "On-Line Analytical Mining Of Association Rules", Master Thesis, Simon Fraser University, Canada.

[80] Gao, W., (2004). "A Hierarchical Document Clustering Algoritm", Master Thesis, Dalhousie University, Halifax, Nova Scotia.

[81] Şeker, Ş. E., Bilgisayar Kavramları, Apriori Algoritması, http://www.bilgisayarkavramlari.com/2011/09/07/apriori-algoritmasi/, Mart 2013.

[82] Machine Learning Group at the University of Waikato, http://www.cs.waikato.ac.nz/ml/weka/documentation.html, Nisan 2013.

[83] Challagulla, V.U.B., Bastani, F.B., I-Ling Yen, ve Paul, R.A., (2005). "Empirical assessment of machine learning based software defect prediction techniques", 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems, Sedona, AZ, USA 263-270.

[84] Han, J. ve Kamber, M., (2006). Data Mining Concepts and Techniques, Morgan Kauffmann Publishers Inc., 1-35.

[85] Cataldo, M., Bass M., Herbsleb J. D. ve Bass L., (2006). "Managing Complexity in Collaborative Software Development: On the Limits of Modularity", Proceedings of Supporting the Social Side of Large Scale Software Development a CSCW Workshop, Banff, AB.

[86] Software Engineering 2004, Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, 23 Agust 2004.

[87] Heil, M.R., (1999). "Preparing Technical Communicators for Future Workplaces: A Model that Integrates Teaming, Professional Communication Skills, and a Software Development Process", Proceedings of the 17th annual international conference on Documentation, SIGDOC, New Orleans, Louisiana, USA.

[88] Rosca, D., (2005). "Multidisciplinary and Active/Collaborative Approaches in Teaching Requirements Engineering", European Journal of Engineering Education, 30(1):121-128.

[89] Pressman, R. S., (2009). Software Engineering: A Practitioner's Approach, 7 Edition, McGraw-Hill.

[90] Highsmith, J. ve Cockburn, A., (2001). "Agile Software Development: The Business of Innovation", Computer, 34(9):120-127.

[91] Grinter R. E., (1995). "Using a Configuration Management Tool to Coordinate Software Development", Proceedings of the Conference on Organizational Computing Systems COOCS'95, Milpitas, California, USA.

[92] Nakakoji, K., Yamamoto, Y. ve Ye Y., (2006). "Supporting Software Development as Knowledge Community Evolution", Proceedings of Supporting the Social Side of Large Scale Software Development a CSCW Workshop, Banff, AB.

[93] Kılıç, Ö., Çağıltay, N. ve Tokdemir, G., (2006). "Yazılım mühendisliği diyagramlarının kullanımındaki bilişsel ve davranışsal özellikler", Zonguldak, Türkiye, 349-355.

[94] Song I. Y. ve Froehlich K., (1995). "Entity-relationship modeling, In: Potentials", IEEE 13, 5:29-34.

[95] Wikibooks, Introduction to Software Engineering, http://en.wikibooks.org/wiki/Introduction_to_Software_Engineering/Print_version, Mart 2013.

[96] Fowler, M. ve Scott, K., (1999). UML Distilled Second Edition A Brief Guide to the Standard Object Modeling Language, Publisher: Addison Wesley, Second Edition.

[97] Kılıçaslan, Y., Nesneye yönelik analiz ve tasarıma giriş, yilmazkilicaslan.trakya.edu.tr/teaching/uml_giris.ppt, Nisan 2013.

[98] IBM, Creating use case diagrams, http://publib.boulder.ibm.com/infocenter/rhaphlp/v7r6/index.jsp?topic=%2Fcom.ibm.rhp.uml.diagrams.doc%2Ftopics%2Frhp_c_dm_use_case_diagrams.html, Nisan 2013.

[99] Yazılım Geliştirmede Sistem Modelleme, http://web.itu.edu.tr/~kanoglu/crs-iscpm-systemmodeling.pdf, Nisan 2013.

[100] Kalıpsız, O., Buharalı, A. ve Biricik, G., (2011). Sistem Analizi ve Tasarımı, Papatya Yayıncılık, İstanbul.

[101] Janzen, D. ve Saiedian, H., (2005). "Test-driven development concepts, taxonomy, and future direction", Computer, 38(9):43-50.

[102] Jantan, H., Hamdan, A. R. ve Hamdan, Z. A., (2009). "Potential Data Mining Classification Techniques for Academic Talent Forecasting", Ninth International Conference on Intelligent Systems Design and Applications, PISA, ITALY.

[103] Delen, D., Walker, G. ve Kadam, A., (2004). "Predicting breast cancer survivability: a comparison of three data mining methods", Artificial Intelligence in Medicine, 34(2):113-127.

[104] Bellaachia, A., ve Guven, E., (2006). “Predicting breast cancer survivability: a comparison of three data mining method”, Ninth Workshop on Mining Scientific and Engineering Datasets in conjunction with the Sixth SIAM International Conference on Data Mining (SDM 2006), Bethesda, MD, USA.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Pınar CİHAN

Doğum Tarihi ve Yeri : 15.03.1986, Diyarbakır

Yabancı Dili : İngilizce

E-posta : pkaya@yildiz.edu.tr

ÖĞRENİM DURUMU

Derece	**Alan**	**Okul/Üniversite**	**Mezuniyet Yılı**
Lisans	Bilgisayar Mühendisliği	Bahçeşehir Üniversitesi	2010
Lise	Fen Bilimleri	Özel Amid Lisesi	2003

İŞ TECRÜBESİ

Yıl	**Firma/Kurum**	**Görevi**
2012-	Yıldız Teknik Üniversitesi	Araştırma Görevlisi
2011-2012	Namık Kemal Üniversitesi	Araştırma Görevlisi

YAYINLARI

Bildiri

1. Cihan, P. ve Kalıpsiz, O., (2013). "Assessing the Human Factors in Software Development Courses Students Project", International Conferance on Education and Educational Technologies, 16-19 July 2013, Rhodes (Rodos) Island, Greece.