
GAZİ ÜNİVERSİTESİ
Gazi
1926

YAPI VE ÇEVRE ÖZELLİKLERİNE GÖRE YIKIM METODU SEÇİMİNDE MAKİNE ÖĞRENMESİ İLE GELİŞTİRİLEN BİR KARAR DESTEK YAZILIMI

İbrahim Cihan YETİŞKEN

DOKTORA TEZİ
ENDÜSTRİYEL TEKNOLOJİ EĞİTİMİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ŞUBAT 2021

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

……..
İbrahim Cihan YETİŞKEN
18/02/2021

YAPI VE ÇEVRE ÖZELLİKLERİNE GÖRE YIKIM METODU SEÇİMİNDE MAKİNE ÖĞRENMESİ İLE GELİŞTİRİLEN BİR KARAR DESTEK YAZILIMI

(Doktora Tezi)

İbrahim Cihan YETİŞKEN

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Şubat 2021

ÖZET

Yapı yıkımı, çok kriterli bir karar verme problemi olarak ele alınmaktadır. Bu problemin çözülmesinde, yapıların ve çevresinin özelliklerine göre yıkım mühendislerinin geçmiş deneyimleri öne çıkmaktadır. Teknik seçimine, uzmanların geçmiş deneyimlerinin yanında ülkemizde uygulanan yıkım yönetmeliklerindeki kriterlerin de göz önüne alınarak karar verilmesi gerekmektedir. Bu tez çalışması kapsamında yıkım teknikleri seçiminde önceki verilerden öğrenebilmesi nedeniyle makine öğrenmesi algoritmaları kullanılmıştır. Bu amaçla kullanılacak olan veri seti önceki çalışmalardan elde edilememesi nedeniyle sentetik olarak üretilmiştir. Veri setinin üretilme çalışmalarında, ülkemizde uygulanan yıkım yönetmeliklerindeki faktörlerle birlikte yıkım uzmanların görüşleri doğrultusunda doğrulanan veriler kullanılmıştır. Bu veriler ile gerçekleştirilen makine öğrenmesi eğitim süreçleri sonucunda bir model oluşturularak, geliştirilen web tabanlı yazılım ile entegre edilmiş ve uzmanlara yıkılmak istenen binaların özelliklerine göre bir yıkım tekniği önerilmiştir. Yazılım ile aynı zamanda bina özellikleri verileri ve uygun yıkım tekniği önerileri uzmanlardan alınmaktadır. Uzmanlardan gelen verilerle de makine öğrenmesi süreçleri işletilerek öğrenme oranı artan bir sistem hedeflenmiştir. Veri setinin gelişmesiyle ülkemizde bu alanda çalışmak isteyen araştırmacılara da yol göstereceği düşünülmektedir.

Bilim Kodu : 92408
Anahtar Kelimeler : Karar destek sistemleri, makine öğrenmesi, yıkım teknikleri, akıllı yıkım tekniği seçimi, süreç tasarımı
Sayfa Adedi : 132
Danışman : Prof. Dr. Abdullah TOĞAY

A DECISION SUPPORT SOFTWARE DEVELOPED WITH MACHINE LEARNING IN SELECTION OF A DEMOLITION METHOD ACCORDING TO STRUCTURAL AND ENVIRONMENTAL FEATURES OF BUILDING

(Ph. D. Thesis)

İbrahim Cihan YETİŞKEN

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

February 2021

ABSTRACT

Building demoliton is considered as a multi-criteria decision problem. Past experiences of demolition engineers come to the fore in solving this problem based on the characteristics of the structures and their surroundings. The selection of technique should be decided by considering the criteria in the demolition regulations applied in our country as well as the past experiences of the experts. Within the scope of this thesis, machine learning algorithms have been used in the selection of demolition techniques because it can learn from previous data. The data set to be used for this purpose was produced synthetically since it could not be obtained from previous studies. In the studies of producing the data set, the factors in the demolition regulations applied in our country and the data verified in line with the opinions of demolition experts were used. As a result of the machine learning training processes carried out with these data, a model was created, integrated with the developed web-based software, and a demolition technique was proposed to experts according to the characteristics of the buildings to be demolished. With the software, building properties data and suitable demolition technique recommendations are received from experts. A system with increasing learning rate is targeted by operating machine learning processes with data from experts. It is thought that with the development of the data set, it will guide researchers who want to work in this field in our country.

Science Code : 92408
Key Words : Decision support systems, machine learning, demolition techniques, intelligent demolition techniques selection, process design
Page Number : 132
Supervisor : Prof. Dr. Abdullah TOĞAY

TEŞEKKÜR

Hayata ve akademisyenliğe bakışım noktasında fikirleriyle, tecrübesiyle bana yol gösteren, her zaman destek olan ve bu çalışmamda da tez danışmanım olarak bana en büyük katkıları veren değerli hocam Prof. Dr. Abdullah TOĞAY' a teşekkürlerimi sunarım.

Çalışmalarım sırasında, fikirleriyle ve tecrübeleriyle bana yol gösteren çok değerli hocalarım Prof. Dr. Serkan GÜNEŞ, Prof. Dr. Ahmet KARAARSLAN, Prof. Dr. Özgür ANIL ve Dr. Öğr. Üyesi Abdullah ORMAN hocalarıma teşekkürlerimi sunarım.

Tez çalışmam süresince, değerli vaktini ayırarak hiçbir zaman yardımlarını esirgemeyen ve bana büyük katkılar sunan kıymetli arkadaşım Dr. İbrahim EDİZ' e teşekkürlerimi sunarım.

Hayatım boyunca her konuda benim yanımda olan, evlatları olmaktan gurur duyduğum annem ve babamla birlikte, çok sevgili kardeşime bu süreçte bana verdikleri sınırsız desteklerden dolayı teşekkürlerimi sunarım.

Burada isimleri geçmese de verdikleri desteklerle tez sürecimde beni motive eden tüm hocalarıma ve dostlarıma teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

RESİMLERİN LİSTESİ

Resim **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
dbA	Decibel-A weighting
PGV	Peek ground velocity

Kısaltmalar	**Açıklamalar**
AHP	Analytic hiyerarchy process
ANP	Analytic network process
CSS	Cascading style sheets
CSV	Comma seperated values
DBMS	Database management system
DGMS	Dialog generation and management system
DTSS	Demolition techniques selection system
HTML	Hyper text markup language
JSON	Java script object notation
KNN	K nearest neighbor
Linear SVC	Linear support vector classifier
MBMS	Model base management system
NoSql	Not only sql
SGD	Stochastic gradient descent
SVC	Support vector classifier
SVM	Support vector machines

Kısaltmalar	Açıklamalar
VTYS	Veritabanı yönetim sistemi
XML	Extensible markup language

1. GİRİŞ

İnsanlık tarihi kadar eski olan barınma ihtiyacının ortaya çıkardığı yapılar insanların yaşamsal faaliyetlerini sürdürebilmeleri, kendilerini ve varlıklarını dış etkilere karşı koruyabilmeleri amacıyla farklı malzemeler ve bu malzemelere uygun yapım teknikleriyle inşa edilen tesislerdir (Togay, 2002). Kentlerdeki nüfus artışına bağlı artan yapılar ve bir Şekilde gelişmekte olan kentlerde sanayileşme, göç, doğal afet gibi farklı sebeplerden dolayı dönüşümler meydana gelmektedir. Ülkemizin deprem kuşağında oluşuna bağlı olarak yaşanan olaylar da kentsel dönüşüm konusunun ön plana çıkmasını sağlamaktadır (Genç, 2008). Söz konusu bu dönüşümler bir taraftan yapıların güçlendirilmesi kapsamında çalışmalar gerektirirken diğer taraftan riskli bulunan yapıların yıkılarak yeniden inşası da gerçekleştirilmektedir.

Öcal ve İnce, 2012'de yaptıkları çalışmada aynı konuyu başka bir açıdan değerlendirmiş, kentsel dönüşümün binaların günümüzdeki ihtiyaçları farklı sebeplerden dolayı karşılayamaması sebebiyle gerçekleştiğini ifade etmiştir. Bu bağlamda, kentsel dönüşüm projeleri, insanların günümüz değişen ihtiyaçlarına uyum sağlama projeleridir. Bu projeler ile bitişik nizam binalardan, site içerisinde konforlu, sosyal alanları olan yaşam alanlarına geçilebilmektedir (Öcal ve İnce, 2012). Görgülü, çalışmasında (2009) doğal afetler kentsel dönüşümleri zorlayan nedenlerden olsa da, uluslararası sermayelerin artışı ve kentlere katkıları, kentlerin girişimci bir hal almasını sağlamıştır demektedir. Kentlerdeki girişimcilik faaliyetleri ile birlikte ekonominin canlanması ve yeni istihdam alanlarının da oluşturulması için de kentsel dönüşümler gerçekleştirilmektedir (Görgülü, 2009).

Resim 1.1. Bitişik nizam yapılar (Öcal ve İnce, 2012)

Gerek doğal gerekçeler gerekse de ekonomik gelişmeler ve yeni ihtiyaçlar kapsamında olsun yapıların da ömrünü tamamlaması sonucu ortadan kaldırılması ve ortaya çıkan alanların yeniden ele alınması kent yaşamının değişmez bir olgusudur. Bu süreç canlı dünyasında olduğu gibi bir yaşam döngüsü içerisinde gerçekleşmektedir ve nihayetinde yapılar, kullanım ömrünün sonuna geldiklerinde ya da herhangi bir Şekilde hasar aldığında yıkılmaları gerekmektedir (Elias Özkan, 2012). Farklı sebeplerden dolayı ortaya çıkan yapı yıkım süreci, birçok aşamadan oluşmakla birlikte, bu aşamalardan en önemlisi, yapıların özelliklerine göre hangi yöntem ile yıkılması gerektiğinin belirlenmesidir. Yıkım teknikleri seçiminde kullanılan yöntemler çok çeşitli olmakla birlikte ülkemizde bu tekniklerin seçiminde kullanılan akıllı karar destek sistemleri ve standart prosedürler bulunmamaktadır. Bu nedenle teknik seçimlerinde uzmanların önceki tecrübelerinden faydalanılmaktadır (Şahmaran, Özgür, Gürbüz ve Koçkar, 2015: 28). Bu durum, uygulamalarda kişiye göre değişen ve bazen de olması beklenen standartları karşılamayan yöntemler ve bu yöntemlerin beklenmeyen sonuçlarına sebep olabilmektedir. Bu uygulamaların yönetmeliklerle sınırları çizilmiş standartlar yanında sürekli yinelenen çalışmaların çıktılarından analitik bir bilgi ile beslenerek kusursuza doğru giden süreçlere dönüşmesi ideal bir tanımlama olarak görülmektedir. Arham tarafından 2003 yılında yapılan çalışmada, yıkım teknikleri seçiminde analitik hiyerarşi prosesi yöntemi kullanılarak, binalar için optimum yıkım tekniği, binaların özelliklerine göre belirlenmiştir. Bu çalışmada kullanılan yöntem gereği, yıkılacak olan binanın özelliklerinin, birbirine göre önem dereceleri karşılaştırılarak, yıkım tekniği karar desteği sağlanmıştır.

Mevcut durum ele alındığında söz konusu alanda yapılmış olan akademik çalışmalar oldukça sınırlıdır ve diğer taraftan yaşanan deneyimler çok sayıda sorunla karşılaşıldığını göstermektedir. Bu durum bu kapsamda tezin çerçevesi ve teknoloji destekli olabilecek yeni bir sürecin tartışılması açısından motivasyon kaynağı olarak değerlendirilmiştir. Buradan hareketle tez kapsamında yıkım süreçlerinin en önemli aşamalarından birisi olan yıkım tekniği seçiminde, makine öğrenmesi algoritmaları temelinde akıllı karar destek sistemi geliştirilmiştir. Bu sistem ile yıkım tekniği seçiminde uzmanlara karar desteği verilmesi amaçlanmaktadır. Bu destek, uzmanlar tarafından yıkılacak olan binanın özelliklerinin ve çevresel koşullarının belirlenerek sisteme girilmesinden sonra sağlanmaktadır. Makine öğrenmesi temelli sistemler için en önemli girdi olan verilerin yıkım uzmanları tarafından sisteme girişinin yapılabilmesi, uygulama sonuçlarına dayalı veriler ile uzmanların dayanarak olarak verdiği ve geçmişte başarılı olan yıkım projelerinde uygulanmış tekniklerin sisteme girilmesi ile baz veri havuzu oluşturulması hedeflenmiştir. Ancak bu veri havuzu geçmişe dair kayıtların sınırlı olması ve alanda yapılmış olan bir çalışmaya rastlanmaması nedeniyle sınırlı bir altlık sunmuştur. Bu çerçevede ülkemizde bu anlamda uygulanmakta olan regülasyonlara uygun ve akademik verilerle de beslenen bir veri setinin daha oluşturularak karşılaştırmalı bir çalışma ile kararlı sonuçlar üretebileceği düşünülmüştür ve sınırları yönetmeliklerle çizilmiş ve uzmanlar tarafından doğrulanmış bir veri seti oluşturularak yıkım süreçlerinin tasarlanmasına yol gösterici olmuştur. Veri setinin oluşturulmasıyla, makine öğrenmesi süreçleri işletilerek, veriler arasındaki ilişkilerin matematiksel hesaplamalarla ortaya çıkarılması sağlanmaktadır.

Makine öğrenmesi süreçleri neticesinde, sistemi daha sonra kullanacak olan uzmanlar, sadece yıkılacak yapıya ait verilerin girişlerini yaparak, kendi projelerine uygun yıkım kararı noktasında destek alabilmektedirler. Makine öğrenmesinin etkinliğinin artmasındaki bir diğer faktör ise, veri setlerinin kararlı verilerle beslenmeye devam etmesini sağlamaktır. Bu kapsamda veri alınması ve verilerin işlenmesi için yazılım desteği önemli görülmüştür. Veri girişlerinin, veri tabanına yapılabilmesi için web tabanlı bir yazılım geliştirilmiştir.

Veri setinin oluşturulmasının ardından, veri setindeki verilerin analizleri yapılarak, bu verilere uygun makine öğrenmesi algoritmaları seçimleri gerçekleştirilmiştir. Bu çalışma için Pedregosa ve arkadaşlarının geliştirdiği makine öğrenmesi algoritması seçimi yol haritasından faydalanılmıştır. Bu yönteme ilişkin bilgiler ileride makine öğrenmesi başlığı altında detaylı olarak sunulacaktır.

Bütün bu çerçeveden hareketle elde edilen kütüphanelerin sağladığı makine öğrenmesi algoritmaları ile yazılım destekli bir sürece taşınma önerisi getirilen bu uygulamaların yeni kararlar alınırken eski karar ve sonuçlarından öğrenen ve başarısı her geçen gün artış göstermesi hedeflenen bir yaklaşımla desteklenmesi ana hedef olarak ortaya konulmuştur.

Bu tez kapsamında yanıtı aranacak sorular şunlardır;

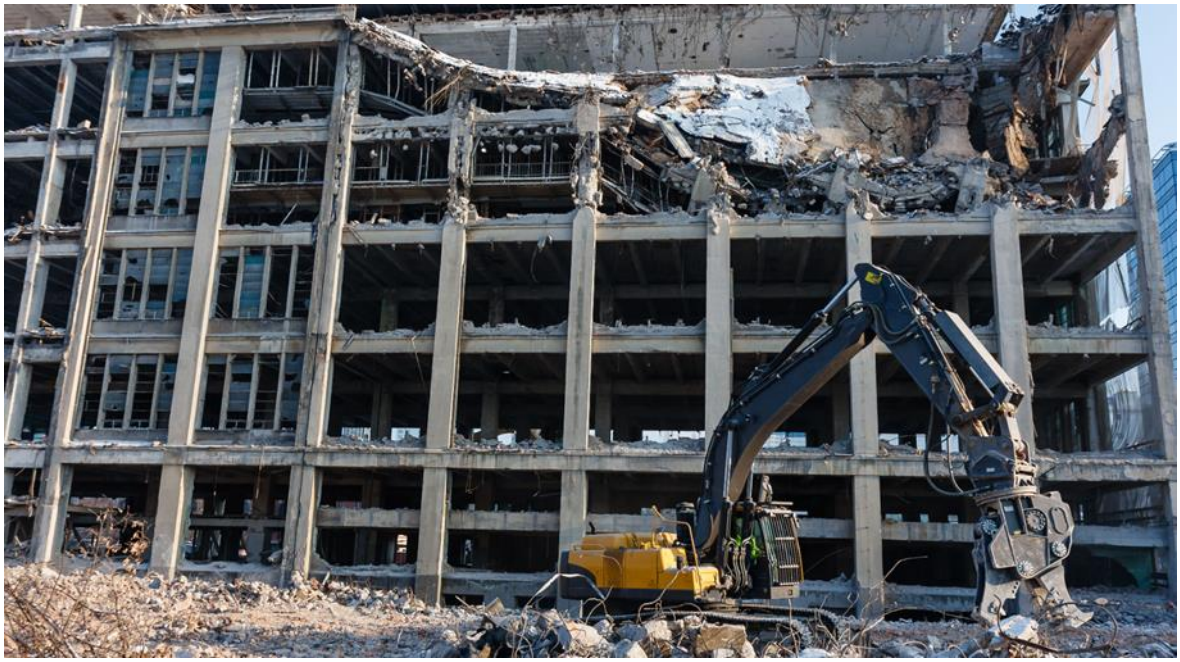
1. Yapı yıkım tekniği seçimine etki eden faktörler nelerdir?
2. Yapı yıkım teknikleri seçiminde yapıların ve bulunduğu çevrenin özellikleri incelenerek, yıkım teknikleri seçiminde makine öğrenmesi yaklaşımlarından faydalanılabilir mi?
3. Yapı yıkım teknikleri seçiminde kullanılabilecek makine öğrenmesi modelleri hangileridir?
4. Yıkım teknikleri seçimi için bir veri seti geliştirilebilir mi ve yıkım uzmanları tarafından veri girişi nasıl olmalıdır?
5. Yapı yıkım süreçlerinin yönetimine dair etkin teknolojik bir yöntem önerisi geliştirilebilir mi?

Sonuç olarak bu tez kapsamında yapı yıkım teknikleri ve bu teknikleri etkileyen faktörler incelenerek, yapı yıkım uzmanlarına teknik seçiminde karar desteği sağlayacak makine öğrenmesi tabanlı bir sistem geliştirilmesine ilişkin yöntemler tartışılmış, denenmiş ve optimal sonuçlar üzerinden öneriler geliştirilmiştir.

2. MEVCUT DURUM VE GENEL BİLGİLER

2.1. Yapı Yıkımı

Ekonomik ömrünü tamamlamış ya da bir Şekilde hasar görmüş yapıların çeşitli teknikler kullanılarak imha edilmesine yıkım denir (Şahmaran ve diğerleri, 2015: 9). Her ne kadar binalar yapılırken sonsuza dek kalacak gibi düşünülüp gelecek nesiller için de yapılıyor olsa da, gerçekte durum tam tersidir. Çoğu bina ömrünü tamamladığında yıkılıp geri dönüştürülmektedir. Bina yıkımında ortaya çıkan malzemeler genellikle yıkım alanından satılmaktadır (Elias Özkan, 2003).



Resim 2.1. Yapı yıkımı (The Balance Small Bussines, 2019).

Ömrünü tamamlayan yapıların yok edilmesine yıkım denir. Ancak yıkım konusunu yok etmekten ziyade, seçici yıkım olarak ele alıp değerlendirmek daha doğrudur. Seçici yıkım ile, malzemelerin kullanılamaz hale getirilip imha edilmesi yerine, geri dönüştürülmesi esas alınmıştır. Yıkım sonrasında ortaya çıkan atıkların da, ülkemizde yürürülükte olan yönetmeliklere uygun olarak değerlendirilmesi gerekmektedir (Abanuz, 2005). Geri yönlü inşa olarak da adlandırılan seçici yıkım, yapıyı oluşturan değerli materyallerin dikkatlice birbirlerinden ayrılması işlemini tanımlamaktadır. Bu sayede seçici yıkım ile ayrıştırılan materyaller yeniden kullanım için değerlendirilebilir.

Modern dünyada yıkım, alanların yeni yapılar için temizlenmesinden, yıkılan yapılardan kazanılan geri dönüşüm malzemelerinin elde edilmesine kadar bir takım karmaşık görevlerin birleşmesinden oluşmaktadır (Diven ve Shaurette, 2010:1).

2.2. Türkiye'de Yıkım

Ülkemizin deprem kuşağında olması nedeniyle meydana gelen depremler, yapılarda hasarlara neden olmuştur. Bu hasarlar, yapıların yıkılıp yeniden inşası sonucunu ortaya çıkarmıştır. Yapılar depremde hasar almamış olsalar bile, kentsel dönüşümler kapsamında depreme daha dayanıklı yapılar inşaa edilmesi ihtiyacı da ülkemiz için karşımızda duran ihtiyaçlardan bir tanesidir. Bu dönüşümlerin yasal düzenlemelerle belirli kriterler gözetilerek yapılması gerekmektedir. Bu sebeple konuyla ilgili yasalarda ilk düzenleme 2005 yılında yürürlüğe giren 5393 sayılı Belediye Kanunun 73. maddesi ile yapılmıştır. Bu madde, kentsel dönüşümlerde belediyelere yetki tanımıştır ancak belediyelerin birbirinden bağımsız olarak bu süreçleri yönetmesindeki çıkabilecek maddi olumsuzlukların ve kısıtlamaların varlığı nedeniyle bu sistemin yerine, merkezi bir koordinasyon ile bu işlemin yapılmasının daha faydalı olacağı düşünüldüğünden, 2011 yılında kurulan Çevre ve Şehircilik Bakanlığı, kuruluş aşamasında kentsel dönüşümü temel hedeflerinden birisi olarak belirlemiştir (Şahmaran ve diğerleri, 2015: 1).

Çevre ve Şehircilik Bakanlığının, hafriyat toprağı, inşaat ve yıkıntı atıklarının kontrolü yönetmeliğinin 19.maddesinde;

> Yıkımı yapılacak yapıların içlerindeki geri kazanılabilir malzemelerin öncelikle ayrıştırılması ve geri kazanılması esastır. Bu çerçevede kapı, pencere, dolap, taban ve duvar kaplamaları, döşemeleri ve yalıtım malzemeleri gibi inşaat malzemeleri ile tehlikeli atıklar yıkımı yapılacak yapılardan ayıklanır ve ayrı toplanır. Yıkım işlemleri sırasında gürültü, toz ve görüntü kirliliği ile ilgili olarak 20 ve 21 inci maddelerde belirtilen tedbirler alınır. Yıkımın hidrolik ekipmanlara sahip iş makineleri ile yapılması durumunda, kolon ve kiriş gibi beton yapılar kesilir veya parçalanır. Çalışanların sağlığını ve güvenliğini korumak amacıyla, asbest içeren malzemelerin kullanıldığı binaların yıkımı, sökümü, tamiratı ve tadilatı sırasında Çalışma ve Sosyal Güvenlik Bakanlığı tarafından hazırlanan ve 26/12/2003 tarihli ve 25328 sayılı Resmî Gazete'de yayımlanan Asbestle Çalışmalarda Sağlık ve Güvenlik Önlemleri Hakkında Yönetmelik esaslarına uyulur (Çevre ve Şehircilik Bakanlığı, 2004).

Aynı yönetmeliği 20. Maddesinde gürültü emisyonları ile ilgili olarak;

“Hafriyat, inşaat/tamirat/tadilat ve yıkım işlemleri sırasında oluşacak gürültü emisyonları ile ilgili olarak, 11/12/1986 tarihli ve 19308 sayılı Resmî Gazete’de yayımlanarak yürürlüğe giren Gürültü Kontrol Yönetmeliği esaslarına uyulur.” denilmektedir.

Yönetmeliğin 21. Maddesinde ise toz emisyonu ile ilgili olarak;

> Hafriyat, inşaat/tamirat/tadilat ve yıkım işlemleri sırasında oluşacak toz emisyonları ile ilgili olarak, 02/11/1986 tarihli ve 19269 sayılı Resmî Gazete’de yayımlanarak yürürlüğe giren Hava Kalitesinin Korunması Yönetmeliği’nde belirtilen hava kalitesi standartlarının sağlanması gerekir. Oluşacak toz emisyonlarının asgariye indirilmesi, görüntü kirliliğinin önlenmesi ve gerekli emniyet koşullarının sağlanması amacı ile tadilat/yıkım yapılacak binaların dış cephesi yırtılmaz ve tutucu özelliğe sahip file ve benzeri malzeme ile koruma altına alınır (Çevre ve Şehircilik Bakanlığı, 2004).

Ülkemizde kentsel dönüşüm ile ilgili kanunlar ve kapsamları aşağıdaki gibidir.

a) 5393 sayılı Kanun’un 73 üncü maddesi kapsamında;

> Belediye, belediye meclisi kararıyla; konut alanları, sanayi alanları, ticaret alanları, teknoloji parkları, kamu hizmeti alanları, rekreasyon alanları ve her türlü sosyal donatı alanları oluşturmak, eskiyen kent kısımlarını yeniden inşa ve restore etmek, kentin tarihi ve kültürel dokusunu korumak veya deprem riskine karşı tedbirler almak amacıyla kentsel dönüşüm ve gelişim projeleri uygulayabilir. Bir alanın kentsel dönüşüm ve gelişim alanı olarak ilan edilebilmesi için yukarıda sayılan hususlardan birinin veya bir kaçının gerçekleşmesi ve bu alanın belediye veya mücavir alan sınırları içerisinde bulunması şarttır. Ancak, kamunun mülkiyetinde veya kullanımında olan yerlerde kentsel dönüşüm ve gelişim proje alanı ilan edilebilmesi ve uygulama yapılabilmesi için ilgili belediyenin talebi ve Cumhurbaşkanınca bu yönde karar alınması şarttır (Belediye Kanunu, 2005).

b) 5366 sayılı Kanun’un 1. maddesi kapsamında;

> Bu Kanunun amacı, büyükşehir belediyeleri, büyükşehir belediyeleri sınırları içindeki ilçe ve ilk kademe belediyeleri, il, ilçe belediyeleri ve nüfusu 50.000'in üzerindeki belediyelerce ve bu belediyelerin yetki alanı dışında il özel idarelerince, yıpranan ve özelliğini kaybetmeye yüz tutmuş; kültür ve tabiat varlıklarını koruma kurullarınca sit alanı olarak tescil ve ilan edilen bölgeler ile bu bölgelere ait koruma alanlarının, bölgenin gelişimine uygun olarak yeniden inşa ve restore edilerek, bu bölgelerde konut, ticaret, kültür, turizm ve sosyal donatı alanları oluşturulması, tabiî afet risklerine karşı tedbirler alınması, tarihi ve kültürel taşınmaz varlıkların yenilenerek korunması ve yaşatılarak kullanılmasıdır (Yıpranan tarihi ve kültürel taşınmaz varlıkların yenilenerek korunması ve yaşatılarak kullanılması hakkında kanun, 2005).

c) 6306 sayılı Kanun’un 1. maddesi kapsamında;

> Bu Kanunun amacı; afet riski altındaki alanlar ile bu alanlar dışındaki riskli yapıların bulunduğu arsa ve arazilerde, fen ve sanat norm ve standartlarına uygun, sağlıklı ve güvenli yaşama çevrelerini teşkil etmek üzere iyileştirme, tasfiye ve yenilemelere dair usul ve esasları belirlemektir (Afet riski altındaki alanların dönüştürülmesi hakkında kanun, 2012).

Ülkemizde yıkım süreçleri sonrası oluşan atıklar, genelde dolgu malzemesi olarak kullanılmaktaydı. Yıkım işleri ile uğraşmakta olan firmalar, yıkım sonrası ortaya çıkan artıkların aslında değerli olduğunu gördüklerinde, yıkım işlemlerini de seçici yıkım teknikleri ile gerçekleştirmeye başlamışlardır (Şahmaran ve diğerleri, 2015: 1). Kentsel dönüşüm regülasyonları nedeni ile ortaya çıkan yıkım ihtiyaçlarının arkasında bırakacağı değerli artık miktarları düşünüldüğünde, seçici yıkım ve buna bağlı yıkım tekniklerinin seçiminin de önemi anlaşılmaktadır.

Türkiye'de bir çok kentsel dönüşüm uygulaması bulunmaktadır. 1995 yılında uygulanan Aydın ili kentsel dönüşümü buna örnektir. Bu dönüşüm, kent merkezinin tamamını içermekte olup, planlama ve uygulama yapısıyla bütünleşik bir model ortaya koymaktadır. Dönüşüm kapsamında, master plan, kentsel tasfiye uygulamaları ve bölgeleri, kentsel yenileme uygulama ve bölgeleri, kentsel ıslah bölgeleri, kentsel tarama ve düzenleme bölgeleri belirlenerek dönüşüm projesi uygulanmıştır. Bu projeler 2002 yılına kadar devam ettirilmiş olup, daha sonra uygulamalara son verilmiştir.

Ülkemizde uygulanan bir başka kentsel dönüşüm projesi ise Afyonkarahisar ilinde gerçekleştirilmiştir. Afyonkarahisar'da uygulanan dönüşümler şehrin büyük bir kısmını kapsamaktansa daha lokal çözümler olarak ortaya konmuştur. Şehrin otogarı'nın yeri değiştirilerek eski otogar yerine yeni lüks yapılar inşa edilmiş, yeni otogarın da şehir dışına taşınarak merkezdeki yoğunluğun azaltılması sağlanmıştır. (Koçak ve Tolanlar, 2008).

Ankara'nın Altındağ ilçesinde geçmişten gelen çarpık yapılaşma ve gecekondular bulunması, bu bölgede suç oranlarının artmasına, hizmetlerin ulaştırılmasında zorluklara ve sağlıksız yaşam alanları ortaya çıkmasına neden olmuştur. Bölgedeki bu yapılaşma ve halkın beklentileri nedeniyle de kentsel dönüşüm çalışmalarında birçok sorunlar ortaya çıkmıştır. Sorunların farklı yöntemlerle çözülmesi ile birlikte, Altındağ ilçesinde çeşitli kentsel dönüşüm yöntemleri uygulanarak bölgenin daha sağlıklı bir hale getirilmesi amaçlanmıştır. Bu yöntemler;

Sokak sağlıklaştırması: Ankara kalesi etrafındaki sokaklar düzenlenerek, evlerin dış cepheleri yenilenmiştir.

Tek yapı ölçeğinde restorasyon: Bölgede bulunan bazı konaklar satın alınarak tamamen restore edilmiştir.

Mülk sahibinin restorasyonu kendisi yapması: Bu yöntem ile belediye alt yapı kaynaklarını hazırlayarak, restorasyonun mülk sahibi tarafından yapılması sağlanmıştır.

Mülk sahipleri ile belediyenin protokol yapması: Bu yöntemde, belediye tüm restorasyon işlerini yaparken, mülk kullanım haklarını belirli sürelerle elinde bulundurmuştur. Süreler sonunda mülk tekrar sahibine devredilecektir.

Koruma amaçlı kamulaştırma: Projede sunulan diğer yöntemleri kabul etmeyen mülk sahiplerinin mülklerinin kamulaştırılarak koruma altına alındığı yöntemdir.

Yukarıda anlatılan yöntemlerle Hamamönü semtinde kapsamlı bir iyileştirme gerçekleştirilmiştir.

Resim 2.2. Hamamönü, 2008 yılından bir görüntü (Sadioğlu, Tiryaki ve Korkmaz, 2016)

Resim 2.3. Hamamönü, 2012 yılından bir görüntü (Sadioğlu ve diğerleri, 2016)

Altındağ ilçesinde gerçekleştirilen bir başka kentsel dönüşüm projesi ise, çinçin mahallesinde gerçekleştirilmiştir. Bölgenin %85'inin gecekondular oluşması, beraberinde birçok sorunu getirmiştir. Bu durum, bölgenin dönüşümü öncelikli hale gelmesine neden olmuştur. Mahallenin dönüşümünde TOKİ tarafından gecekondular yıkılarak yerlerine yüksek katlı binalar inşa edilmiştir. Bu noktada gecekondu hak sahiplerinin daire sahibi olmaları da hedeflenmiştir (Sadioğlu, Tiryaki ve Korkmaz, 2016).

Resim 2.4. Çinçin eski halinden bir görüntü (Sadioğlu ve diğerleri, 2016)

Resim 2.5. Çinçin kentsel dönüşümden sonraki halinden bir görüntü (Sadioğlu ve diğerleri, 2016)

Depremler nedeniyle de birçok kentsel dönüşüm projeleri gerçekleştirilmiştir. İstanbul'da tarih boyunca birçok deprem meydana gelmiş ve yapılar yerlerinde yenilenmiştir. Meydana gelen doğal afetler ile aynı zamanda şehirlerdeki imar planları da yenilenmiştir. Ülkemizde meydana gelen bazı depremlerden sonra, yerleşim yerlerinin yerleri değiştirilerek daha

güvenli alanlarda inşa edilmişlerdir. 1939 yılında gerçekleşen Erzincan, 1970 yılında Gediz, 1942 yılında Erbaa, 1975 yılında Lice, 1939 yılında Dikili depremlerinden sonra yerleşim yerleri daha güvenli bölgelere taşınmıştır. Ülkemizde 1999 yılında meydana gelen Marmara depremi, büyüklüğü, etkilediği alan açısından bakıldığında, yenileme ve risk azaltma yöntemleri bakımından ülkemizde bu konuda bir dönüm noktası olmuştur (Genç, 2008).

Örnekler incelediğinde, ülkemizdeki kentsel dönüşüm uygulamalarının yaygınlığının, gerek doğal afetler gerekse de gelişen ihtiyaçlar nedeniyle artış gösterdiği gözlenmektedir. Bu anlamda bakıldığında, yapı yıkımlarının sistematik olarak geliştirmesi gerektiği düşünülmektedir.

2.3. Dünyada Yıkım

Amerika ve Kanada'da yürütülen yıkım faaliyetleri Avrupa, Japonya ve birçok ülkede gerçekleştirilen yıkım faaliyetlerinden aşağıdaki nedenlerden dolayı farklılıklar göstermektedir;

- Yasal düzenlemeler
- Teknolojik ekipmanların varlığı
- İşçilik maliyetleri
- Asbest gibi zararlı maddeler içeren bileşenlerin kaldırılması için gerekli yasal düzenlemelerin değişkenliği
- Yıkım sonrası ortaya çıkan malzemelerin değerleri ve pazarlanabilirliği
- Geri dönüşüm prosedürleri konusundaki farklılıklar (Yılmaz, Çankaya ve Karakaya, 2017).

2.4. Yıkım Teknikleri

Binaların yıkımında teknik seçimleri, yıkım maliyetleri açısından da önem arz etmektedir. Yıkım maliyetlerinin düşürülmesi için, bina yıkımında doğru tekniklerin seçilmesi önemlidir. Örneğin bitişik nizam bir bina yıkılmak istendiğinde, patlayıcı ile yıkım yerine küçük iş makineleri ile yıkım daha doğru bir tercih olacaktır (Yılmaz ve diğerleri, 2017). Yıkım teknikleri birçok faktöre bağlı olmakla birlikte, önemli faktörler aşağıdaki gibidir;

- Yıkılacak yapının tipi
- Yıkılacak yapının boyutları

- Yıkım için kullanılacak ekipmanların mevcudiyeti, olmayanlar için kiralama durumu.
- Gereksinimlerin belirlenmesi
- Kısıtlamalar
- Personel durumu
- Müteahhitin iş yükü (Diven ve Shaurette, 2010:13).

Her yapının yıkım tekniği, yıkım parametrelerinin fazlalığından dolayı kendisine özgüdür. Bir parametre, bir yıkım için çok önemli iken, farklı bir yıkım için daha az önemli ya da önemsiz olabilir. Örneğin yapılar birebir aynı bile olsa, farklı konumlarda olduklarında yıkım teknikleri de değişiklik gösterebilir. Bu gibi farklılıklar, uygun yıkım teknikleri seçiminde yıkım mütehaitinin işini zorlaştırmaktadır. Bu nedenlerle, yıkım tekniklerini formülize etmek mümkün değildir ancak bazı anahtar ilkeler belirlenebilir (Anumba, Abdullah, ve Fesseha, 2003).

Yapı yıkımında yapının kısmi ya da tamamen yıkılması arasında farklar bulunmaktadır. Deprem gibi doğal afetler nedeniyle hasar almış yapıların kısmi olarak yıkılmasında yapının ana taşıyıcı kolonlarının korunması beklenmektedir. Bu yıkım türünde çevreye mümkün olduğunca az zarar verecek Şekilde yıkım tekniklerinin uygulanması gerekmektedir. Bu nedenle detaylı planlama yapılması gerektiğinden maliyet de önemli oranda artmaktadır. Yıkım teknikleri temel olarak aşağıdaki başlıklarda belirtilmiştir;

- Mekanik ekipmanlar ile parçalanarak yıkım
- Çelik bir küre yardımıyla yapıya vurarak parçalama yoluyla yıkım
- Mekanik ekipmanlar yardımıyla ayırarak yıkım
- Patlayıcılar ile yıkım
- Kimyasal malzemelerin genleşme özelliğini kullanılarak yapının parçalanarak yıkılması
- Elmas testereler ile yapının elemanlarının kesilmesi yoluyla yıkım.

Bir yapının yıkımında bir çok teknik kullanılmaktadır. Bu teknikler arasından seçim yapmak için aşağıdaki kriterler göz önünde bulundurulmaktadır;

- Yıkım maliyeti
- Zaman sınırlamaları
- Yapının betonarme taşıyıcı elemanlarının kalitesi
- Yapının geometrisi
- Yapının konumu

- Yapının çevre özellikleri
- Yapının türü ile ilgili özel riskler
- Yıkım sonrası oluşan atıkların yeniden kullanımları
- Ulaşım özellikleri

Bu kriterler arasında en önemlileri maliyet ve yapının boyutları olarak öne çıkmaktadır. Diğer kriterler, bu iki kriterin durumuna göre Şekillenmektedir. (Koca, 2006).

2.5. Genel Bilgiler

Bu bölümde tezin motivasyon kaynağı olan yıkım teknikleri konusunda geliştirilecek makine öğrenme tabanlı bir karar destek sistemi için kullanılan teorik girdiler başlıklar halinde açıklanmıştır.

2.5.1. Karar verme süreçleri

Karar verme, farklı çözüm alternatifleri arasından hedefe uygun olan çözüm alternatifini seçme sürecidir (Forman ve Selly, 2001:1). Karar verme süreçleri, hayatın her alanında karşı karşıya kaldığımız bir durumdur. Çok kriterli karar verme aşamalarında ise, kriterlerdeki artış, karar verme süreçlerini zorlaştırmaktadır. Bu gibi çok kriterli karar verme süreçlerinde, sayısal yöntemlerin kullanımının süreçlere olumlu etkisi bulunmaktadır (Carlson ve Fuller, 1996).

Çok kriterli karar verme yöntemleri, ölçülebilen ve ölçülemeyen bir çok faktörü aynı anda değerlendirme imkanı sağlayan ve bu süreçlere bir çok kişinin dahil olabilmesini sağlayan analitik yöntemlerdir. Bu yöntemler ile karar vericilerin daha doğru karar alabilmesi sağlanarak işletme maliyetlerine de olumlu etkileri bulunmaktadır (Dağdeviren, Eraslan, Kurt ve Dizdar, 2005).

Temel anlamda karar verme, bir problemin çözümü için sunulan çeşitli alternatifler arasından seçim yapmak anlamına gelmektedir (Druzdzel ve Flynn, 2011:462). İnsanların karar vermesine yardımcı olabilmek için Karar Destek Sistemleri kullanılır;
Karar destekleri sistemleri aşağıdaki sebeplerden dolayı kullanılmaktadır

1- Hızlı hesap yapabilme: Karar sürecinde hız önemlidir. Bu nedenle karar destek sistemi, karar vericinin daha hızlı karar vermesine olanak sağlamaktadır.
2- Geliştirilmiş iletişim: Karar verme sürecinde birden fazla karar verici olduğu durumlarda, web tabanlı uygulamalar ile karar vericiler arasındaki iletişim güçlenmektedir.
3- Verimlilik artışı: Daha güçlü iletişim olanakları nedeniyle yer zaman fark etmeksizin karar vericilerin iletişimi sağlayarak verimliliği arttırmaktadır.
4- Farklı veri kaynaklarının kullanımı: Karar verme esnasında, karar destek sistemine yüklenmiş olan text, görsel, video gibi farklı veri kaynaklarından gelen verilerin işlenmesi ile etkili karar verme sağlanmaktadır.
5- Farklı veritabanlarına erişim: Problemin çözümüne yönelik karar verme sürecinde farklı veritabanlarından gelecek olan verileri de işleme olanağı bulunmaktadır.
6- Karar kalitesinin artması: Bilgisayarlar, farklı kaybaklardan gelen verilerin hepsini aynı anda işleyebildiğinden, verilen kararın kalitesi de artmaktadır (Turban, Jay ve Ting-Peng, 2007: 10-11).

2.5.2. Karar destek sistemleri

Temel anlamda karar verme, bir problemin çözümü için sunulan çeşitli alternatifler arasından seçim yapmak anlamına gelmektedir. Karar destek sistemleri, insanlara karar verme ve seçim yapma konusunda yardımcı olan bilgisayar tabanlı yazılımlardır. Karar destek sistemlerinin üç temel öğesi bulunmaktadır;

1- Veritabanı yönetim sistemi (DBMS) : Probleme dayalı verilerin saklandığı alanlardır. Buradaki veriler analiz edilerek, karar vericiye yardımcı olunmaktadır.
2- Model tabanlı yönetim sistemi (MBMS) : Veritabanı yönetim sistemindeki verilerin anlamlı bir Şekilde modellendiği katmanlardır. Bu katman, veritabanındaki yapılar ile kullanıcı etkileşiminin olacağı yapı arasında bulunmaktadır.
3- Dialog oluşturma ve yönetim sistemi(DGMS): Karar destek sistemi ve kullanıclar arasında etkileşimi sağlayan katmandır. Kullanıcıların bilgisayar bilgisi olmasına gerek kalmaksızın, karar destek sistemini kullanabileceği özellikleri, kullanıcılara sunmaktadır (Druzdzel ve Flynn, 2011: 465).

Belirsizlik seviyesi yüksek olan karar verme problemlerinde, karar vericiye analitik modeller kullanarak karar vermesinde yardımcı olan sistemlere, karar destek sistemi denmektedir. Karar destek sistemleri aynı zamanda bir bilgi sistemidir (Çetinyokuş ve Gökçen, 2002).

Çok kriterli karar verme problemleri bir çok alanda sıklıkla karşımıza çıkmaktadır. Bununla beraber, karar verme süreçlerinde insanların öznel davranışları karar verme sürecini olumsuz etkilemektedir (Xiao, 2020).

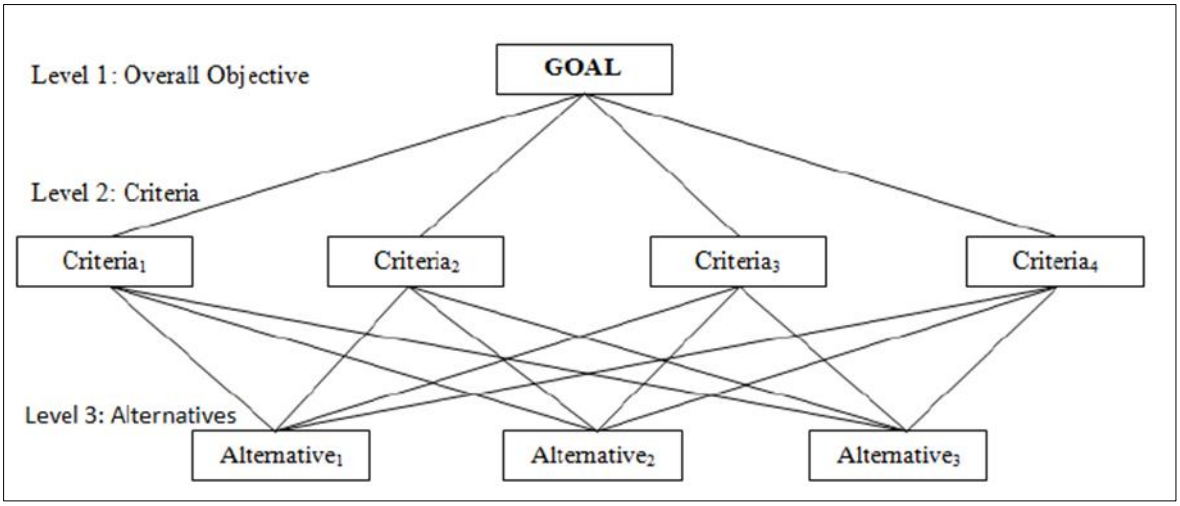
Birden çok kriter içeren problemlerde, çok kriterli karar verme analizlerini kullanmak, karar vericilere yardımcı olmaktadır. Çok kriterli karar verme analiz yöntemleri, probleme sistematik bir Şekilde yaklaşarak sonuca giden yolda karar vericilerin işlerini kolaylaştırmaktadır (Cables, Lamata ve Verdegay, 2016).

2.5.3. Analitik hiyerarşi süreci / Analytic hierarchy process (AHP)

Karar verme süreçlerinde kriterlerin birbirine göre önemini sayısallaştırıp matrisler aracılığı ile karşılaştırarak, doğru kararın verilmesine yardımcı olan yöntemdir. Bu yöntem Thomas L. Saaty tarafından 1970'lerde bulunmuştur. Esnek ve kolay kullanımlı olması nedeniyle çok kriterleri karar verme problemlerinin bir çoğunda kullanılmaktadır.

Çok kriterli problemlerde karar verme aşamasına gelindiğinde, insanlar mantık yerine duygularıyla hareket edebilmektedir. AHP ile kriterler sınıflandırılır ve her bir kriterin birbirine göre öncelikleri matrislerde sayısal olarak ifade edilir. Bu Şekilde kriterler arasında karşılaştırma yapılarak karar ortaya çıkar (Rohland, 2017).

AHP yöntemi, çok kriterleri karar verme süreçlerinde, nitel ve nicel değişkenlerin aralarındaki ilişkileri ve önceliklerini bir arada değerlendirebilen matematiksel bir yöntemdir. Bununla birlikte bu yöntemin kolay uygulanabilir olması, yönteme olan ilgiyi arttırmaktadır (Orman, Düzkaya, Hayri ve Akdemir, 2018).



Şekil 2.1. AHP'nin hiyerarşik yapısı (Agarwal, Sahai, Mishra, Bag ve Singh, 2014)

2.5.4. Makine öğrenmesi

Makine öğrenmesi her geçen gün genişleyen bir perspektifte birçok yeni alanda kullanılmaktadır. Bu kapsamda makine öğrenmesi konusunu ele alan çok kapsamlı bir bilgi havuzu mevcuttur. Tez çalışmasında ise bu kapsamdan teze kaynaklık eden makine öğrenmesi aşamaları ve modelleri ele alınmıştır.

Bilgisayarların icadından bu yana, araştırmacılar insan öğrenmesinin makineler tarafından nasıl yapılacağı sorusu üstünde çalışmalar yapmaktadırlar. Bu çalışmalar ile yapay zeka sistemleri geliştirilmektedir. Öğrenme sürecinin, çeşitli algoritmalar ile bilgisayar tarafından modellenerek oluşturulması ise, makine öğrenmesinin konusudur (Michalski, Carbonell, ve Mitchell, 1983: 25-28).

Makine öğrenmesi, bilgisayarların özel bir işlem için programlanmasına gerek kalmadan öğrenmeleri sağlayan bir sistemdir. Makinelerin öğrenmesi, matematiksel modellerin, veri ile kombine edilmesi neticesinde sağlanmaktadır. Makine öğrenmesinin giriş parametreleri verilerdir. Makine öğrenmesi aşamalarından birisi olan eğitim aşamasında, elimizdeki giriş verilerine bazı matematiksel modeller uygulanarak, veriler arasındaki ilişkiler ortaya çıkarılmaktadır. Bu işlemin çıktısı ise öğrenmiş matematiksel modeldir (Köksal, 2020).
Makine öğrenmesi ve diğer yapay zeka teknikleri, gerçek zamanlı karar verebilmek için geliştirilmiş tekniklerdir ve verilerin yoğun olduğu durumlarda kullanımı artmaktadır.

Verilerin az olduğu durumlarda ise gerçek uygulamalar yerine daha çok araştırma amacıyla kullanılmaktadır (Dixit, Chinnam ve Singh, 2020).

Makine öğrenmesinin amacı, deneyimlerden öğrenerek kendi ortamlarına adapte eden sistemler geliştirmektir. Makine öğrenmesi, bilgisayar bilimleri, mühendislik, matematik, fizik, nöro bilim ve bilişsel bilimler gibi çeşitli alanları içermektedir. Bu alanlardaki araştırmacılar için geniş bir araştırma alanı sunmaktadır (Soman, Loganathan ve Ajay, 2009: 1).

Makine öğrenmesi, görevlerin nasıl yapılaması gerektiği konusunda, bilgisayarlara örnekler verilerek onların öğrenmesini sağlayan araçlar bütünüdür.

Makine öğrenmesinin gerekliliğini farklı açılardan tanımlamak gerekirse;

Yapay zeka açısından makine öğrenmesi: Öğrenme, zeki makineler yapmak için gerekli bir olgudur. Yapay zeka üstünde yıllardır yapılan çalışmalar, karar vermede tüm ihtimalleri düşünerek yapılan programlamanın yetersiz olduğunu ortaya koymuştur. Bu noktada otomatik öğrenmenin önemi çok büyüktür. Örneğin, insanlar doğduklarında herhangi bir dil bilmezken, zaman içerisinde öğrenmektedirler. Makinelerde de dili tüm yönleriyle programlamaktansa, kendi kendine öğrenmesi sağlamak çok daha etkili bir yol olmaktadır.

Yazılım mühendisliği açısından makine öğrenmesi: Makine öğrenmesi, bilgisayarlara örnekler göstererek kod yazmayı sağlayabildiğinden, klasik kodlama yöntemlerine göre daha kolay olabilmektedir.

Programlama, kapsamlı bilgi ve deneyim gerektiren zor bir görevdir. Makine öğrenmesi yöntemleri ile programlamanın yeniden Şekillendirilmesi mümkündür. Doğal dil işleme gibi bazı alanlarda makine tarafından öğrenilen algoritmalar, el ile geliştirilen algoritmaların yerini almaya başlamıştır. Bunun çok daha ilerilere götürülmesi mümkündür. Makine öğrenmesi yöntemleri ile öğrenilen algoritmalar ile daha etkin otomatik algoritmalar geliştirilerek, güvenli, efektif ve doğru algoritma geliştirme maliyetlerinin de düşeceği söylenebilir (Gottschlich, Solar-Lezama, Tatbul, Carbin, Rinard, Barzilay, Amarasinghe, Tenenbaum ve Mattson, 2018).

İstatistiksel açıdan makine öğrenmesi: Makine öğrenmesi, bilgisayar bilimleri ve istatistiğin ortak bir çalışma alanıdır. Bilişim ile ilgili problemler, istatistiksel problemlere uygulanır. Makine öğrenmesi, istatistiksel problemlerin ötesinde birçok probleme uygulanabilir. Örneğin hızın, doğruluktan daha önemli olduğu durumlarda kullanılabilir.

Makine öğrenmesi ile hayvanların öğrenmesi benzerlik göstermektedir. Örneğin fareler bir yiyeceğin kokusunu aldıklarında önce küçük bir parça ile tadına bakmaktadırlar. Bu deneme sonrasında iyi hissederlerse ya da tadını beğenirlerse yemeye devam ederek, bu yiyeceği "yenilebilir" olarak etiketlemektedirler. Daha sonra tekrar aynı yiyecekle karşılaştıklarında ise önceki tecrübelerinden faydalanıp, buna göre hareket etmektedirler.

Bu mantıktan hareketle, temel bir makine öğrenme senaryosunu örneklemek gerekirse, spam e-posta'ların ayrıştırlması örneği verilebilir. Bir e-posta'nın spam olup olmadığına makinenin karar vermesi isteniyorsa, spam olan e-postaların, "spam" olarak etiketlenmesi gerekmektedir. Bu etiketleme işlemi, kullanıcılar tarafından yapılmalıdır. Sistem, yeni bir e-posta geldiğinde önceki etiketlemelere bakarak spam olup olmadığına karar verecek, eğer posta daha önce spam olarak etiketlenmişse, spam klasörüne, değilse kullanıcının gelen posta kutusuna yönlendirilecektir.

Yukarıdaki örnek, temel olarak başarılı olsa da, sürekli olarak maillerin kullanıcılar tarafından etiketlenmesi gerekliliği nedeniyle kısıtlılıkları bulunmaktadır. Fareler örneğinde, yiyecekten bir örnek olarak tadına bakan ve onu "yenilebilir" olarak etiketleyen fare, daha sonra farklı ancak benzer tat ve kokusu olan yiyecekle karşılaştığında onu da "yenilebilir" sınıfına koyabilmektedir. İyi öğrenen bir makinenin de benzer Şekilde hareket etmesi gerekmektedir. Sistem, bir mail geldiğinde kullanıcı spam olarak etiketledikten sonra, etiketlenen mailin içerisinde geçen kelimeleri ve cümle yapılarını da kaydetmesi gerekmektedir. Bu sayede, gelecek olan maillerin de spam olup olmadığına, içeriklerine bakarak karar vermesi sağlanabilir. Bu yöntem, tümevarım yaklaşımı olarak da düşünülebilir (Shalev-Shwartz ve Ben-David, 2014: 21-22).

Makine öğrenmesi aşamaları

Makine öğrenmesi problemleri çözümünde aşağıdaki aşamalar kullanılır.

Veri hazırlama

Makine öğrenmesinde kullanılacak verilerin hazırlandığı aşamadır.

Depoloma

Nosql veritabanları

NoSql veritabanları, veritabanı yönetim sistemleri (VTYS) ailesinin yeni bir üyesi sayılmaktadır. Nosql veritabanları, büyük veri saklama ve analizleri için oldukça uygundur. İlişkisel veritabanlarında olan şema kavramı bulunmadığı için daha ölçeklenebilir bir yapıdadır (Frozza, Jacinto ve Mello, 2020).

Json

Json, günümüzde kabul edilen ve geçerli veri değişim, gösterim biçimidir. Karmaşık yapılardaki verilerin hızlı bir Şekilde gösterimini ve transfer edilmesini sağlar (Frozza ve diğerleri, 2020). Json veri biçimi, dinamik web uygulamaları ve temel veri transferi işlemleri için oldukça uygundur. Json biçimdeki veriler, Xml biçimindeki verilere göre daha hızlı transfer edilir. Bunun sebebi, json biçiminin daha basit yapısı ve bu sayede veriye erişimin hızlı olmasından kaynaklanmaktadır (Darmawan, Rahmatulloh, Nuralam, Rianto ve Gunawan, 2020).

Csv

Virgülle ayrılmış değerler (Comma seperated values) anlamına gelen csv, web üstünde veri paylaşma noktasında kullanılan önde gelen formatlardan bir tanesidir. Bu nedenle, csv formatındaki veriler, uygulamalar arasında veri transferi yapmak için kullanılmaktadır (Mahmud, Hossin, Jahan, Noori ve Bhuiyan, 2018).

Xml

XML, Extensible Markup Language kelimelerin baş harflerinden oluşan bir veri saklama dosya biçimidir. Xml biçiminde saklanan veriler XPath ve XQuery ile sorgulanabilmekte ve sıralanabilmektedirler (Saito, Yuyama, Ishii, Huang ve Nishi, 2020). Xml, farklı kaynaklar arasındaki veri alış verişinin standartlaştırmak için kullanılmaktadır. Farklı veri kaynaklarını kullanarak kurumsal düzeyde bir veri entegrasyon çözümü uygulamak ve sunmak için kullanılan bir teknolojidir (Jayashree ve Priya, 2020).

Toplama

Web scraping,web crawling

Web sayfaları, HTML dili ile oluşturulmuştur. Sayfalarda bulununan içeriklerse HTML etiketleri arasında yer almaktadır. Web scraping işlemi, bu içerikleri HTML etiketleri arasından okuyarak, çevrimiçi fiyat karşılaştırma, ürün yorumları karşılaştırma, hava durumu takibi gibi çeşitli uygulamalarda bu verilerin kullanılmasını sağlar (Uzun, 2020).

Veri seti arama motorları

Veriler, bilimsel araştırmacılar, analiz uzmanları, gazeteciler gibi dünyanımızı daha iyi anlamak isteyen bir çok kişi için oldukça önemlidir. Çoğu veri, web'de hükümetler, bilimsel yayıncılar, ticari veri sağlayıcıları, araştırma konsorsiyumları tarafından yayınlanmaktadır. Bu Şekilde web'de bir çok veri kaynağı bulunmaktadır (Brickley, Burgess ve Noy, 2019). Veri seti arama motorları ile web üzerinde bulunan bu kaynaklar içerisinde arama yapmak mümkün olmaktadır.

Üretme

Sentetik veri setlerinin üretimi

Sentetik veri setleri, gerçek veriler toplanmadan üretilen, uygun makine öğrenmesi modelinin belirlenebilmesi için kullanılan veri setleridir. Sentetik veri setlerinin üretiminde beklenen özellikleri şunlardır;

- Veriler nümerik, mantıksal veya kategorik olabilir.
- Üretilecek verilerin özellik sayıları ve veri setinin uzunluğu isteğe bağlı olmalıdır.

- Veriler rastgele olarak üretilmeli ancak kullanıcı aralıkları belirleyerek kontrol edebilmeli
- Veriler bir sınıflama algoritması için üretilecekse, öğrenme problemini zor ya da kolay yapabilmek için sınıfların ayrılık dereceleri kontrol edilebilir olmalıdır
- Veriler arasına rastgele gürültülü veri kontrol edilebilir Şekilde eklenebilir
- Regresyon problemi için, karmaşık, doğrusal olmayan veri üretim süreci kullanılmalıdır (Sarkar, 2018).

Problem cümlesinin hazırlanması

Gerçek dünya uygulamalarında, uygulamanın gereksinimlerini belirlemek ve hangi kriterlerin değerlendirileceğine karar vermek çok önemlidir. Sorunun çözümünün etkili olabilmesi ve doğru karar verebilmek için gereksinimlerin açıkça belirlenmesi gerekmektedir (Rahman, Sungyoung ve Tae, 2017). Yıkım teknikleri seçiminde de bir çok kriter bulunmakla birlikte, bu kriterler ve sonuca nasıl ettikleri konuları daha sonra incelenecektir.

Problem cümlesine uygun makine öğrenme yaklaşımının tespiti

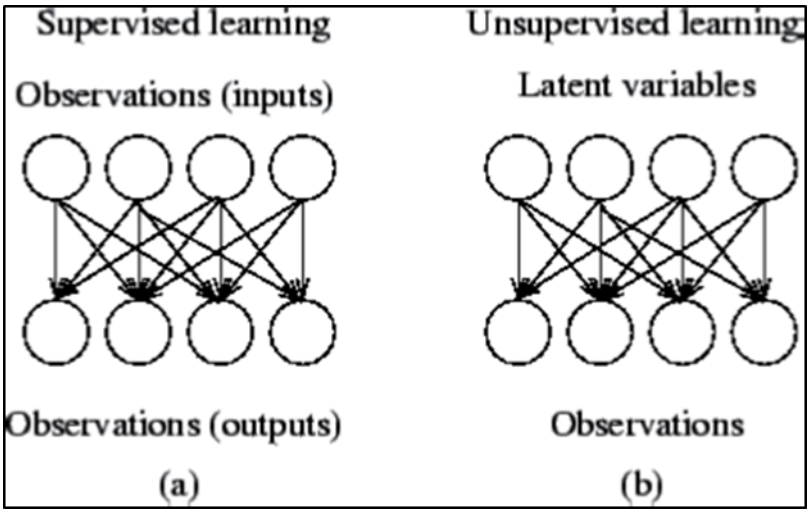
Makine öğrenmesi, öğrenme yöntemlerine göre genel olarak 3 ana başlıkta incelenebilir. Bu tez kapsamında problemin niteliğine uygun olarak denetimli öğrenme algoritmaları kullanılmıştır;

1. Denetimli Öğrenme
2. Denetimsiz Öğrenme
3. Derin Öğrenme

<u>Denetimli öğrenme</u>

Denetimli öğrenme, genel olarak sınıflandırma problemlerinde kullanılmaktadır. Burada hedef, yapılan sınıflandırma sonucu bilgisayarın öğrenmesini sağlamaktır. Denetimli öğrenmede, giriş parametreleri ve çıkış parametreleri bulunmaktadır. Giriş parametrelerinden alınan değerler ile, çıkış parametrelerine ulaşmak için fonksiyonlar geliştirilmektedir. Bu öğrenme metodunda hem giriş hem de çıkış parametrelerinin ne olduğu bilinmektedir (Oladipupo Ayodele, 2010: 20).

Denetimli öğrenme süreçlerinde, giriş verileri kullanıcı tarafından verilerek, bu verilerden ulaşılacak olan sonucun ne olduğu etiketlenir. Örneğin giriş verileri olarak, oda sayısı, semt, cephe vb. bilgiler, çıkış bilgileri olaraksa bu evlerin fiyatları olabilir. Bu Şekilde bir eğitim seti tablo halinde hazırlanıp sisteme verilir ve eğitilmesi ile giriş ve çıkış bilgileri arasındaki ilişkilerin öğretilmesi amaçlanır. Daha sonra benzer Şekilde hazırlanan test verileri ile sistemin etkinliği ölçülebilir.



Şekil 2.2. Denetimli ve denetimsiz öğrenme (Oladipupo Ayodele, 2010: 20)

Denetimli makine öğrenmesi yönteminde, etiketlenmiş veriler aracılığı ile eğitim gerçekleştirilir. Etiketlenmiş veri, kriterlerin değerlerine göre, aslında bilinen sonucun etiketlenmesi anlamına gelmektedir. Eğitim aşamasından sonra, denetimli öğrenme algoritmasının eğitimdeki analizlere dayalı çıktılar ürettiği yeni bir veri seti kullanılmaktadır. Denetimli öğrenme yöntemleri temel olarak iki başlık altında incelenebilir. Bunlar, sınıflandırma ve regresyondur. Sınıflandırma problemlerinde denetimli öğrenme çıktısı kategorik verilerken, regresyon problemlerinde öğrenme çıktıları gerçek verilerdir (Köksal, 2020).

Her yapının farklı özellikleri bulunmaktadır. Bu özelliklerden bazıları, boyutları, konumu, komşu yapılara uzaklığı gibi özelliklerdir. Yapı yıkım tekniği seçiminde ise bu özelliklere göre karar verilmekte olup, her yapı için o yapıya özel bir yıkım tekniği belirlenmektedir. Yapıların özelliklere atanan değerler makine öğrenmesi açısından giriş verileri, özelliklerin değerine seçilen yıkım tekniği ya da teknikleri ise çıkış verilerini oluşturmaktadır. Yıkım

teknikleri seçimi bu perspektifte değerlendirildiğinde makine öğrenmesi açısından denetimli öğrenme yöntemleriyle çözümlenebilmektedir.

Denetimli öğrenme modelleri iki başlık altında incelenebilir. Bunlar;

a. Sınıflandırma
b. Regresyon

Sınıflandırma

Sınıflama, belirli özelliklerdeki bir örneğin, hangi sınıfta olduğunun tahmin edilmesidir. Örneğin bir kuşun kanat açıklığı, ayaklarının tipi ve rengine göre kuşun hangi türden olduğuna karar vermek bir sınıflandırma problemi olarak gösterilebilir (Harrington, 2012:8-9). Bu tez kapsamında yapılan çalışmada da bir sınıflandırma problemi bulunmaktadır. Yapıların özelliklerine göre yıkım teknikleri değişkenlik gösterdiğinden, sınıflandırılan tekniklere ulaşmak için sınıflandırma algoritmalarından faydalanılmıştır. Literatürde birçok sınıflandırma algoritması bulunmasına karşın, bu algoritmalar her veri seti için uygun olmamaktadır. Bu tez kapsamında yapılan ve 5.1. bölümde anlatılan veri seti oluşturma aşamasından sonra, elde edilen veriler ile aşağıdaki sınıflandırma algoritmalarının kullanılması kararlaştırılmıştır. Doğru algoritmanın seçimi konusundaki detaylar, 5.2.1. bölümünde ele alınmıştır.

Bu tez kapsamında kullanılan sınıflandırma algoritmaları ve yöntemleri şunlardır;

- Doğrusal destek vektör makineleri (Linear support vector machines – Linear svc),
- K en yakın komşu algoritması (Kneighbors algoritması),
- Destek vektör sınıflandırıcı (Support vector classifier – Svc)
- Olasılıksal dereceli azalma yöntemi (Stochastic gradient descent- Sgd),
- Çekirdek yaklaşımı yöntemi (Kernel approximation)

Doğrusal destek vektör makineleri (Linear support vector machines – Linear svc)

Doğrusal destek vektör makineleri (DVM), kontrollü bir sınıflandırma algoritma olup temelleri istatiksel öğrenme yöntemlerine dayanmaktadır. DVM başlarda iki sınıflı problemler için tasarlanmış olsa da, sonradan çok sınıflı ve doğrusal olmayan problemler için de kullanılabilir hale getirilmiştir. Destek vektör makineleri ile sınıflandırmada iki sınıfa

ait etiketlenmiş verilerin birinden ayrılması amaçlanmaktadır. Verileri birbirinden ayırmakta kullanılan düzleme, hiper-düzlem denilmektedir. DVM, temelde kendisine en yakın noktalar arasındaki uzaklığı maksimuma çıkaran hiper düzlemi bulmayı amaçlamaktadır. Bu hiper düzleme optimum hiper düzlem, sınır genişliğini sınırlandırmak üzere belirlenen vektörelere ise destek vektörleri denilmektedir (Kavzoğlu ve Çölkesen, 2010).

Linear support vector classifier, support vector classifier'ın bir tipi olmakla birlikte, kernel parametresinde linear değeri bulunmaktadır. Bu nedenle, çok sayılı örneklerde daha performanslı ölçeklendirme yapmaktadır (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot ve Duchesnay, 2011).

Linear Svc terimi, açık kaynaklı bir makine kütüphanesi olan python temelli sckit-learn'ün, linear support vector machines için verdiği isimdir.

K en yakın komşu algoritması (Kneighbors algoritması)

K-en yakın komşu algoritması (KNN), sınıflama işlemini gerçekleştirebilmek için veriler arasındaki uzaklıkları kullanır. KNN, özellik uzayındaki veri noktaları arasındaki uzaklıkları ölçmek için, veri noktalarının sayısal ya da çok boyutlu vektörlerden oluştuğunu varsaymaktadır. Birbirine mesafe olarak yakın noktaların benzer özellikler taşıdığı temeline dayanmaktadır. KNN ile veri noktaları arasındaki uzaklıkları hesaplarken en sık kullanılan yöntem, euclidean distance yöntemidir. Bu yönteme göre, bütün veri noktaları özellik uzayının bir vektörüdür ve etiketlerle bu verilerin sınıfları belirlenmektedir. Bu noktada en basit durum, etiketlerin ikili sayı sisteminde olmasıdır ancak farklı Şekilde sayısallaştırılmış etiketler de kullanılabilir. KNN etiketlenmiş verilerden oluşan sınıflandırma problemlerinde kullanılabilir. Etiketlenmiş veriler, eğitim verileri ve test verileri olmak üzere ikiye ayrılmaktadır (Keramati, Jafari-Marandi, Aliannejadi, Ahmadian, Mozaffari ve Abbasi, 2014).

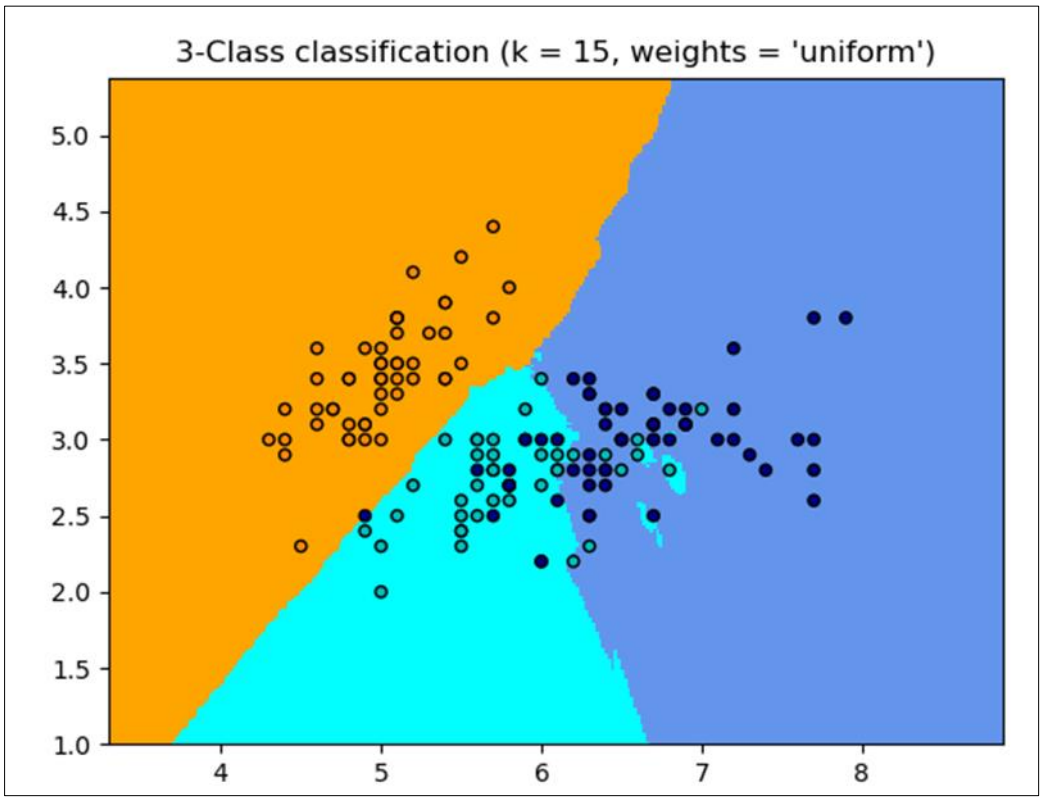
K en yakın komşu algoritmasında, sınıflandırılmak istenen veri olan t için, t'ye en yakın olan k kadar komşu alınarak sınıflandırma işlemi yapılmaktadır. Bu nedenle belirlenecek olan k değeri oldukça önem taşımaktadır. K değerinin doğru olarak belirlenebilmesi için, veri seti

üstünde farklı değerler ile denemeler yapılarak, en iyi sonucu veren değerin seçilmesi gerekmektedir.

K en yakın komşu sınıflandırma algoritmasının bazı karakteristik özellikleri şunlardır;

- Örnek tabanlıdır
- Ezberci öğrenme
- Parametrik olmayan bir tekniktir
- Eğitim için zaman gerektirmemektir.

KNN, eğitim veri setini ezberlemesi nedeniyle tembel bir öğrenme algoritmasıdır (MurtiRawat, Panchal, Singh ve Panchal, 2020).

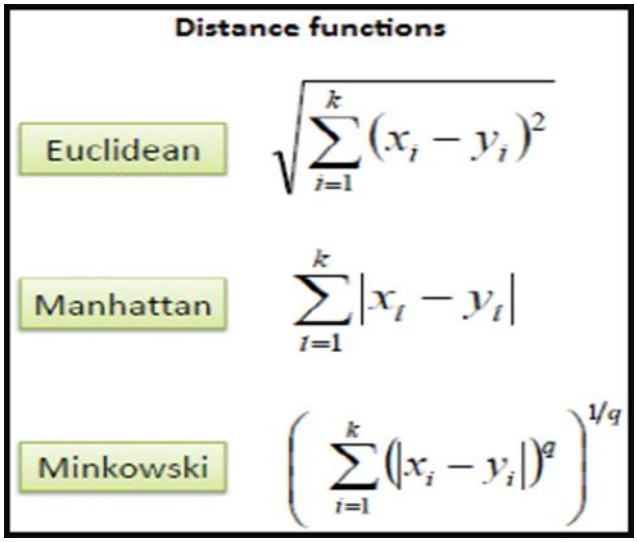


Şekil 2.3. 3 sınıflı k en yakın komşu sınıflandırması (Pedregosa ve diğerleri, 2011)

Şekil 2.4'de gösterilen k en yakın komşu algoritmasında sürekli değişkenler arasındaki uzaklık ölçümü için kullanılan üç adet yöntem bulunmaktadır. Bu yöntemler, euclidian, manhattan ve minkowski yöntemleridir.

K en yakın komşu sınıflandırma algoritmasının çalışma mantığı;

- K değeri seçiminde veri setindeki kayıt sayısının karekökü alınır. K çift sayı olursa 1 çıkarılır.
- Eğitim örnekleri ve sorgu örneği arasındaki mesafe hesaplanır.
- Hesaplanan uzaklıklar, artan sırada sıralanarak en yakın komşular bulunur.
- Seçilen en yakın tüm komşuların sınıfları dikkate alınır
- Sınıfı tahmin edilmek istenen veriye en yakın komşuların ait olduğu sınıf, sorgulanan verinin sınıfı olarak tahmin edilir.



Şekil 2.4. K en yakın komşu algoritmasındaki uzaklık ölçme yöntemleri

K en yakın komşu algoritmasının olumlu yönleri;

- KNN veri ayrımı yapmamakla beraber, doğrusal olmayan veriler için daha kullanışlıdır.
- Anlaşılması kolay ve doğruluk oranı yüksektir
- Hem sınıflandırma hem de regresyon problemlerinde kullanılabilir.

Olumsuz yönleri;

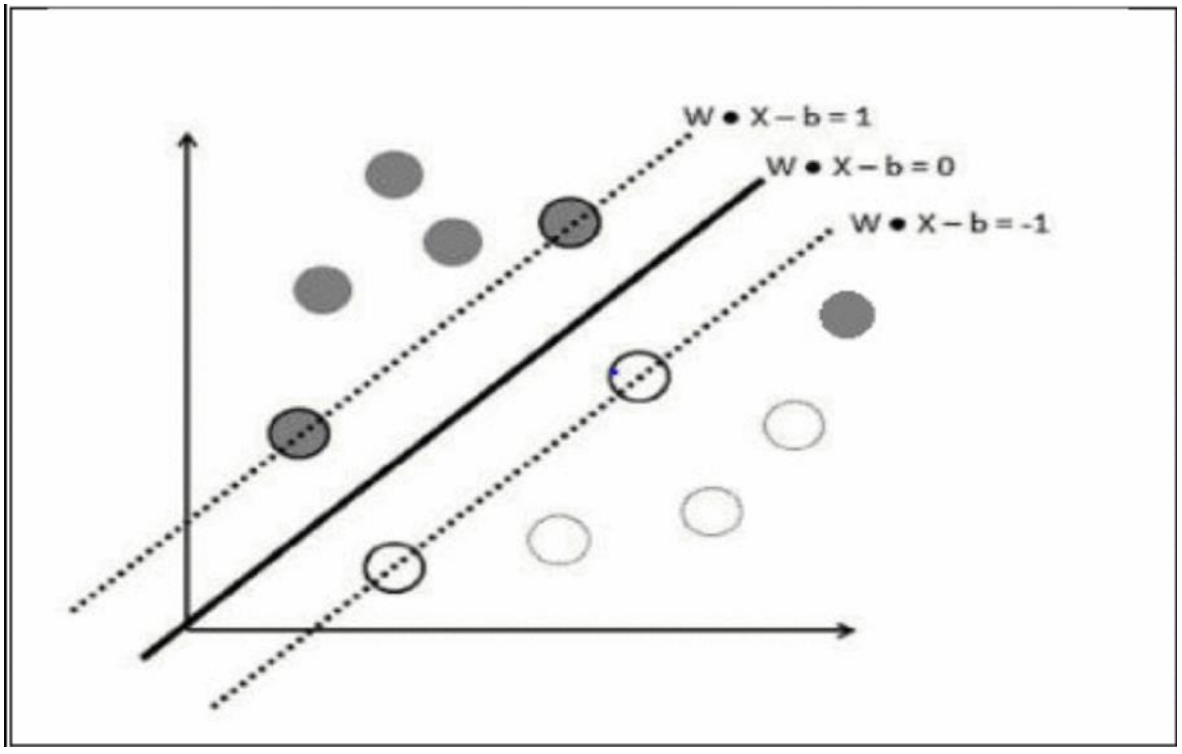
- Tüm eğitim verilerini bellekte sakladığından, yüksek sistem kaynakları kullanımı gerektirir.
- Tahmin etme hızı düşüktür (Bhatnagar ve Srivastava, 2019).

Destek vektör sınıflandırıcı (Support vector classifier – Svc)

Support vector classifier'ın baş harflerinden oluşan svc, açık kaynak kodlu scikit-learn kütüphanesindeki svm yani support vector machines makine öğrenmesi algoritmasının ismidir.

Makine öğrenmesi literatüründe sınıflandırma problemlerinin önemi büyüktür. Farklı sektörlerde sınıflandırma problemlerinin çözümü, son yıllarda makine öğrenmesi alanında önemli bir çalışma alanı haline gelmiştir. Bu anlamda geliştirilmiş en başarılı makine öğrenmesi sınıflandırma algoritmalarından birisi de destek vektör makineleridir. Bu algoritma ile birçok makine öğrenmesi sınıflandırma problemi başarıyla çözülerek literatürdeki yerini almıştır (Ayhan ve Erdoğmuş, 2014).

Destek vektör makineleri (DVM) algoritması, veriyi analiz etme ve veri seti içerisindeki veri desenlerini veya karar sınırlarını belirlemekte kullanılır. DVM, birincil olarak sınıflandırma problemlerinde kullanılmakla beraber, regresyon problemlerinde de kullanılmaktadır. DVM, veri setindeki veriler arasında çok boyutlu düzlemler ile verileri birbirinden ayırarak sınıflandırma işlemini gerçekleştirmektedir. Ortaya çıkan her boyuta, veri setinin özellik vektörü adı verilmektedir. Şekil 2.5' de görüldüğü üzere, DVM birden çok sürekli ve kategorik veri üstünde işlem yapma yeteneğine sahiptir.



Şekil 2.5. DVM sınıflandırıcısı (Burges, 1998)

Şekil 2.5'de içi dolu ve boş olmak üzere iki tip daire gözükmektedir. DVM'in buradaki amacı, iki daire tipini sınıflandırmaktır. Burada üç adet vektör bulunmaktadır. Bunlardan w.x-b=0 olan vektöre sınır vektörü denilmektedir. Diğer iki vektör ise iki farklı sınıfın birbirine en yakın olduğu noktaları göstermektedir. Bu iki vektörün üzerindeki daireler, destek vektörleri olarak isimlendirilmektedir. Diğer sınıfta kalan içi dolu daire ise görmezden gelinmektedir. DVM'in amacı, destek vektörleri arası dikey uzaklığı maksimize etmektir (Ghosh, Dasgupta ve Swetapadma, 2019).

DVM, 1995 yılında Vapnik tarafından geliştirilen, veri madenciliği alanında güçlü ve etkili algoritmalardan bir tanesidir. DVM aynı zamanda maksimal kenar sınıflandırıcısı olarak da bilinmektedir. DVM'in teorik temelleri istatistikten gelmektedir ve makine öğrenmesini de de kapsamaktadır. DVM, eğitilmiş veri setlerinden oluşturduğu giriş ve çıkış eşleşme fonksiyonlarından yaptığı incelemelerle öğrenme işlemini gerçekleştirmektedir (Akman, Karaman ve Kuzey, 2020).

Olasılıksal dereceli azalma yöntemi (Stochastic gradient descent- Sgd)

Olasılıksal dereceli azalma yöntemi, bir makine öğrenmesi algoritması olmasından çok bir optimizasyon tekniği olarak tanımlanabilir. Bu tez kapsamında, doğrusal destek vektör makineleri algoritması ile eğitilmiş model üzerinde optimizasyonlar yapmak amacıyla kullanılmıştır.

Stochastic Gradient Descent sınıflandırıcısı, svm ve lojistik regresyon gibi konveks kayıp fonksiyonları altında, doğrusal sınıflandırıcı ve regresörlerle eğitim için basit ama çok etkili bir yöntemdir. SGD, geniş ölçekli makine öğrenmesi problemlerinde, metin işleme ve doğal dil işleme problemlerinde başarıyla kullanılmıştır. Seyrek verilerin olduğu durumlarda, bu modeldeki sınıflandırıcılar, problemleri 10^5 eğitim verisi örneği ve 10^5'den fazla özellik için ölçeklendirebilmektedir. Net olarak bakıldığında, SGD bir optimizasyon tekniğidir ve herhangi bir makine öğrenmesi modeli kapsamına girmemektedir. SGD, bir makine öğrenmesi modelinin eğitimi için kullanılan bir yöntemdir.

SGD'nin avantajları;

- Verimlilik
- Kolay uygulanabilirlik

Dezavantajları;

- SGD, hiperparametreler ve iterasyonların sayısını istemektedir.
- Ölçeklendirme işlemi konusunda hassastır (Pedregosa ve diğerleri, 2011).

Çekirdek yaklaşımı yöntemi (Kernel approximation)

Bu alt modül, örneğin destek vektör makineleri (DVM) için kullanılan belirli çekirdeklere karşılık gelen veri özellikleri eşlemelerine yaklaşan fonksiyonlar içermektedir. Bu özellik fonksiyonları, giriş verilerinin doğrusal olmayan dönüşümlerini gerçekleştirmekte ve bu, doğrusal sınıflandırma veya diğer algoritmalar için bir temel oluşturmaktadır. Bu fonksiyonların kullanımı, büyük veri setlerinin eğitimlerinde kullanılan kaynakların daha efektif kullanımının yanında daha etkili bir eğitim de sağlamaktadır (Pedregosa ve diğerleri, 2011).
Çekirdek yaklaşımı yöntemi, genelde destek vektör makineleri gibi çekirdek tabanlı algoritmaların büyük veriler üstünde ölçeklendirilmesinde kullanılmaktadır (Cortes, Mehryar ve Talwalkar, 2010). Kernel metodları, yüksek performanslı öğrenme makineleri oluşturmayı amaçlamaktadır ancak, veri sayısı arttığında bazı limitler de ortaya çıkmaktadır (Ionescu, Popa ve Sminchisescu, 2017).

Çekirdek yaklaşımı hesaplamalarında kullanılan bir yöntem olan nystrom yöntemi, birçok makine öğrenimi probleminde ortaya çıkan çekirdek matrislerinin düşük dereceli yaklaşımlarını oluşturmak için popüler bir tekniktir. Nystrom yönteminin yaklaşık kalitesi, büyük ölçüde seçilen sınır noktalarının sayısına ve seçim prosedürüne bağlıdır. (Pourkamali-Anaraki, Becker ve Wakin, 2018). Diğer bir yöntem olan fourier ise, bellek kullanımı açısından maliyetli çekirdek yaklaşımı yöntemini hızlandırırken aynı zamanda doğrusal olmayan öngörü gücünü koruyabilen yeni bir tekniktir (Băzăvan, Li ve Sminchisescu, 2012).

Bu çerçeveden bakıldığında çekirdek yaklaşımları veri sayısı arttığında modelin öğrenme verimliliğini arttırmada kullanılan bir yöntem olarak tanımlanmaktadır. Bu tez kapsamında

çekirdek yaklaşımı kullanan modellerden olan DVM modelinin üstünde denenerek bir sonuç elde edilmiştir.

Şekil 2.5'de destek vektör makinelerinin verileri sınıflandırmada kullandığı teknik gösterilmiştir. Destek vektör makinelerinde sınıflar arasında doğrusal bir vektör ile ayrım yapılmaktadır ancak her zaman bu ayrım sınıfları birbirinden doğru bir Şekilde ayırt edememektedir. Bu noktada çekirdek yaklaşımı kullanılmaktadır. Çekirdek yaklaşımı ile verilerin sınıflandırılması için farklı bir boyut elde edilerek sınıflandırma verimliliğinin arttırılması hedeflenmektedir.

3. LİTERATÜR ÖZETİ

Bu bölümde tez kapsamına kaynaklık edecek nitelikte ve özellikle yıkım teknikleri seçimi ile ilgili alanda yapılmış sınırlı sayıdaki akademik çalışmalar özetlenmiştir.

Arham, 2003 yılında yapmış olduğu doktora çalışmasında yıkım tekniklerinin seçimi probleminin çözümüne dair bir karar verme sistemi önerisi tartışmıştır. Arham'a göre yıkım yıkım teknikleri seçimi, çok kriterli bir karar verme problemidir. Bu problemin çözümünde kullanılan yaklaşımların geliştirilmesi gerekmektedir. Bu yaklaşımlar genellikle yıkım mühendislerinin önceki tecrübelerine dayanarak verdiği kararlardan oluşmaktadır ve sistematik olarak geliştirilmiş bir sistem kullanılmamaktadır. Bunun yanında yıkım teknikleri seçiminde deneyimli karar vericilerin de kararlarını emekli olmadan önce elde edebilmek de önemlidir. Bu amaçla geliştirilen sisteme DTSS (Demolition Techniques Selection System) adı verilmiştir. Geliştirilen sistem temel olarak iki aşamadan oluşmaktadır. İlk aşama, yıkım teknikleri seçiminde karar vericilere yardımcı olabilmek için kurgulanan aşamadır ve bu aşamada AHP modeli (Analitik Hiyerarşi Proses) kullanılmıştır. İkinci aşamada ise yıkım maliyetlerinin belirlenmesi hedeflenmiş ve Yıkım Maliyeti Tahmin modeli geliştirilmiştir. Geliştirilen sistem ile sektöre yeni giriş yapmış genç mühendislerin eğitimine ve yıkım uzmanlarının verdiği kararlara destek olunması hedeflenmiştir (Arham, 2003).

Yıkım, yapım işinin geri dönüştürülmesidir. Yıkım sırasında yapıyı oluşturan bazı bileşenlerin sökülmesi ve geri dönüştürülmesi gerekebilir. Yapı yıkımlarında teknik seçimi yapabilmek birçok kriteri göz önüne alarak en uygun tekniği belirlemek gerekmektedir. Bir yıkım projesi için önemli olan bir kriter, başka bir yıkım projesi için önemli olmayabilir. Yıkım teknikleri seçiminde değerlendirilen temelde altı kriter ve bir çok alt kriter bulunmaktadır. Temel olan altı kriter, yapısal özellikler, çevresel durumlar, yıkım maliyeti, geçmişteki tecrübeler, zaman ve potansiyel geri dönüşümdür. Bir başka çalışmada ise sekiz kriter önerilmiştir. Bunlar, binanın yapısal özellikleri, yapının ölçeği, yapının konumu, izin verilen rahatsızlık düzeyi, yıkımın çerçevesi, binanın kullanım durumu, güvenlik ve yıkım süreci olarak ele alınmaktadır. Bütün bu kriterler, yıkım tekniği seçiminde ele alınmakla beraber, uzmanların ortak olarak üstünde anlaştıkları nokta, güvenlik ve sağlık olmaktadır. Bu açılardan bakıldığında, yıkım tekniği seçiminin, çok kriterli bir karar verme problemi

olduğu sonucu ortaya çıkmaktadır. Bu çalışmada, AHP yöntemi ile yıkım tekniği seçimi gerçekleştirilmiştir. Kriterlerin yıkım projesinin özelliklerine göre ikili olarak önem derecelerinin belirlenmesi ile yıkım tekniği kararı verilmektedir (Anumba, Abdullah ve Ruikar, 2008).

Çok kriterli karar verme yöntemlerinden olan analitik hiyerarşi süreci, karar problemi hiyerarşik olarak yapılandırılmaktadır ve sürecin devamında kriterlerin birbirlerine göre önem dereceleri belirlenmektedir (Orman ve diğerleri, 2018). Bir başka çok kriterli karar verme yöntemi olan analitik ağ süreci ise, Saaty tarafından geliştirilen ve hiyerarşideki öğeler arasında daha karmaşık, birbirine bağımlı ilişkiler ve geri bildirim sağlayan genel bir AHP biçimidir (Sipahi ve Timor, 2010).

Yıkım yöntemlerine karar vermede kullanılan, çok kriterli karar verme yöntemlerinden birisi olan AHP yöntemi ile verilen kararların, mühendislik uygulamalarında uygun olmadığı düşünülmektedir. Bu çalışmada, daha önce AHP ile yıkım kararı verilen bir köprünün yıkım projesinin, ANP ile de değerlendirilerek, AHP ve ANP arasındaki karşılaştırılması yapılmıştır. AHP yerine, ANP yani Analytic Network Process ile karar vermenin, daha keskin sonuçlar ürettiği görülmüştür. Çalışmada, yıkım yöntemleri seçiminde üç aşamalı bir sistem geliştirilmiştir. Birinci aşamada, değerlendirme kriterleri arasındaki ilişkiler belirlenmiştir. İkinci aşamada ise uzmanlara uygulanan anketlerden elde edilen ikili karşılaştırma verilerinin niteliksel olarak tanımlandığı aşamadır. Kriterlerin ikili karşılaştırmaları Super Decisions isimli bir yazılımla gerçekleştirilmiştir. Üçüncü aşamada ise ANP hesaplamalarının sonuçlarından sonra en uygun yıkım planının seçildiği aşama olmuştur. Sonuçlar incelendiğinde, ANP tekniği ile daha doğru kararlar verilebildiği görülmüştür (Chen, Abdullah, Anumba ve Li, 2014).

Yıkım için birçok teknik bulunsa da, yıkılmak istenen yapının özelliklerine ve çevresine uygun yıkım tekniği seçimi oldukça önemlidir. Yıkım tekniği seçiminde dikkat edilmesi gereken kriterler, maliyet, kısmi ya da tamamiyle yıkım, mekanik ekipman yeterlilik düzeyi, yıkılmak istenen yapıda kullanılan malzemelerin kalitesi, yapının geometrisi, yapının boyutları ve konumu, çevresel faktörler ve trafik durumu, yapının bulunduğu noktadaki zemin özellikleri, yıkım tecrübesi, sağlık ve güvenlik önlemleri, araç gereç tedariği, yıkım sırasında ortaya çıkacak zararlı maddelere karşı güvenlik önlemi, izin verilen rahatsızlık düzeyi, yıkımdan sonra ortaya çıkan molozların yeniden kullanımı. Bu kriterlerin arasında

en önemli üç kriter ise, konum, boyutlar ve yıkım maliyetidir. Yıkım işlemlerinin tecrübesiz kişiler tarafından gerçekleştirilmesi, istenmeyen sonuçlar doğurabilmektedir (Tavsan ve Türker , 2020).

4. ARAŞTIRMA YÖNTEMİ

Yıkım süreçlerindeki en önemli aşamalardan olan yıkım tekniği seçimine yönelik sorunların tespiti ve optimal süreçlerin tasarımına odaklanan çalışma 2 aşamalı bir süreçte gerçekleştirilmiştir. Çalışma kapsamında ilk olarak yıkım süreçleri için karar mekanizmaları için yönetmelikler ve senaryolar üzerinden veri setleri oluşturulmuş, ikinci aşamada ise makine öğrenmesi tabanlı teknolojiler üzerinden yürütülen testler üzerinden optimal uygulama sonuçlarına erişilmeye çalışılmıştır.

Tez kapsamında kullanılan yönetmelik verileri dışında olası senaryoların oluşturulması örnek veriler üzerinden işletilmiştir. Yıkım süreçlerinde gerçekleştirilmiş çok sayıda örnek olmakla birlikte karar süreçleri ve kriterler üzerinden bir veri tabanı söz konusu değildir. Bu kapsamda deneyim temelli olmakla birlikte sınırlı sayıda veriye erişilmesi mümkün olmuştur ve hedeflenen makine öğrenmesi temelli karar destek sistemi için yeterli görülmemiştir.

4.1. Verilerin Toplanması

Tez kapsamında sınırlı olduğu tespit edilen veri havuzunun genişletilebilmesi amacıyla olası senaryolar üzerinden gerçekçi sentetik veri üretme işlemleri gerçekleştirilmiştir. Bu kapsamdaki yıkım tekniği kararları verilirken gerekli olan değişkenler, ülkemizde uygulanmakta olan regülasyonlar ve literatür dikkate alınarak belirlenmiştir. Regülasyonlara dayalı değişkenler Bölüm 4.3 içerisinde detaylı olarak sunulacaktır. Bu değişkenlere gelmesi muhtemel değerler ve olası senaryolar, yıkım uzmanları ile görüşülerek elde edilmiş ve sentetik veri üretimi aşamasında bu görüşmelerden çıkan sonuçlar kullanılmıştır.

Bu amaçla deprem mühendisliği alanında çalışan sektörde deneyimli 10 profesyonelle görüşmeler gerçekleştirilmiş, yapılan görüşmelerde yarı yapılandırılmış görüşme formu (EK-6) kullanılmıştır. Görüşmeler akademisyenlerle ele alınmış, 5 akademisyen ve görüşme süreci firma tespiti, iletişim kanalları ile uzman kişinin belirlenmesi, planlı görüşme şartlarının tanımlanması ve görüşmenin gerçekleştirilmesi süreçlerini içerecek Şekilde her bir görüşme ilk tur için minimum 3 saat olacak ve gerekli görülen durumlarda tekrar eden Şekilde 2 aylık süreç içerisinde tamamlanmıştır. Her bir görüşme sonucunda elde edilen kayıt dokümanları çözümlenerek nicel ve nitel değişken matrisleri elde edilmiştir.

4.2. Verilerin Analizi

Tez kapsamında görüşülen uzmanlar aracılığı ile elde edilen görüşme sonuçları ve açık uçlu değerlendirme kayıtları çözümlenerek referans değerler ele alınmıştır. Yapılan değerlendirmelerde ortaya çıkan çelişkili alanlar ikinci tur değerlendirmelerle gözden geçirilerek temiz bir veri seti için altlık sağlanmış ve veri seti üretme prosedürü netleştirilmiştir. Son olarak üretilen veriler ve belirlenmiş yıkım tekniği seçimi kararları tekrar uzman/akademisyen görüşüne sunulmuş ve doğrulamasının yapılması, gerekli yerlerde ise düzeltmelerinin yapılması sağlanmıştır. Bu noktada üretilen verilerin, gerçek verilerle uyumlu maksimum çeşitlilikte olması hedeflenmiştir. Böylece geliştirilen yazılımın fiili durumlarda olma ihtimali düşük değerlerle de çalışabilen, böylece olası tüm senaryolara karşı eğitilmiş bir sistem ortaya çıkarmaktır.

4.3. Yıkılacak Yapıların Verilerinin Toplanacağı Veri Setinin Ülkemizdeki Regülasyonlara Göre Oluşturulması

Yapı yıkım tekniği seçiminde yapıların özellikleri, konumu, çevresel faktörler önemli rol oynamaktadır. Yapı yıkım prosedürleri tasarlanırken, bu faktörlerin hepsinin birden değerlendirilerek teknik seçiminin yapılması gerekmektedir. Bu faktörlere ilişkin toplanması gereken veriler aşağıdaki Çizelge 4.1’ de belirtilmiştir (Şahmaran ve diğerleri, 2015: 34).

Çizelge 4.1. Yıkım tekniği belirlenmesi için gerekli olan veriler

Veri Başlığı	Veri ile ilgili Açıklama	Veri
Yapı Özellikleri	Yapı yüksekliği (metre)	
	Yapı taşıyıcı sistem türü (Betonarme, yığma, çelik, ahşap, mühendislik hizmeti almamış gecekondu, karma)	
	Yıkımın kapsamı (Tümden, kısmi yıkım, acil yıkım)	
	Yapının kullanım şekli (Konut, üretim tesisi, hastane, ibadethane, okul, kütüphane, tehlikeli madde deposu, nükleer tesis)	
	Yapının stabilitesi ve hasar durum (Stabil, yapı hasarlı[ağır,orta,az])	
Saha Özellikleri	Yapının konumu (Tek ayrık, köşede bitişik nizam, her iki tarafından bitişik nizam)	
	Yapının komşu yapılara uzaklığı (metre)	
	Yapının yakınında hastane, ibadethane, okul gibi çok sayıda insanın kullandığı bir tesis olup olmadığı	
	Yapı yakınında alt yapı (su,doğalgaz,ileitşim hattı) veya metro tüneli bulunup bulunmadığı	
	Saha içindeki ve dışındaki insanların sağlığı ve güvenliği	
	Kabul edilebilir rahatsızlık seviyesi	
	Gürültü seviyesi (dBA)	
	Toz seviyesi	
	Titreşim	
	Yıkım sahasına erişim durumu	
Yıkım Maliyeti	Yapının yıkım maliyeti içerisinde seçilen tekniğe göre işçilik, makine ve patlayıcı maliyeti ve toplam maliyet değeri	
Tecrübe	Yıkım firmasının seçilen yıkım tekniğine aşinalık derecesi(iyi, orta, düşük)	
	Yıkım firmasının mevcut personel, tesis ve ekipman mevcudiyet dereceleri (var veya yok var ise iyi, orta, düşük seviye)	
	Yıkım firmasının uzman olduğu yıkım teknikleri (El, makine veya patlatmalı yıkım. Birden fazla ise hangileri olduğu belirtilmeli)	
Geri Dönüşüm	Yıkım firmasının geri dönüşüm uygulaması yapıp yapmadığı, yaptı ise yaptığı işlerdeki geri dönüşüm oranı	
Süre	Yıkım projesinin hedeflenen süresi	

4.3.1. Kategorik veri tiplerinin sayısallaştırılması

Çizelge 4.1'deki veriler incelendiğinde sürekli ve süreksiz karma veri tiplerinden oluştuğu görülmektedir. Bu veri tiplerinin makine öğrenmesi algoritmaları tarafından yorumlanıp aralarındaki ilişkilerin hesaplanabilmesi için süreksiz kategorik verilerin de sayısallaştırılması gerekmektedir. Bu çerçeveden hareketle verilerin sayısallaştırma işlemleri gerçekleştirilmiştir. Sayısallaştırma işleminde, her bir kategorik verinin karşılığına bir değer ataması yapılmıştır.

Yükseklik

Yükseklik verisi metre cinsinden sürekli bir veri olduğundan ve sayısal olarak girileceğinden dolayı herhangi bir düzenleme yapılmamıştır.

Yapı taşıyıcı sistem türü

Yapı taşıyıcı sistemleri Çizelge 4.1' de betonarme, yığma, çelik, ahşap, mühendislik hizmeti almamış gecekondu ve karma olarak kategorilendirilmiştir. Bu kategoriler metinsel ifadeler olduğundan, sayısal olarak temsil edilmeleri gerekmektedir. Bunun için Çizelge 4.2' de gösterilen sözlük tablosunda belirlenen değerler kullanılmıştır.

Çizelge 4.2. Yapı taşıyıcı sistem türü kategorilerinin sayısal olarak temsil edilmeleri

Yapı Taşıyıcı Sistem Türü	Sayısal Karşılığı
Betonarme	1
Yığma	2
Çelik	3
Ahşap	4
Mühendislik hizmeti almamış gecekondu	5
Karma	6

Yapının kullanım şekli

Çizelge 4.1 incelendiğinde kullanım yapının kullanım Şekilleri, konut, üretim tesisi, hastane, ibadethane, okul, kütüphane, tehlikeli madde deposu, nükleer tesis olarak kategorilere

ayrıldığı gözükmektedir. Bu noktada metinsel olan bu ifadelerin de sayısallaştırılması işlemi Çizelge 4.3 ’de gösterilmiştir.

Çizelge 4.3. Yapının kullanım şekli kategorilerinin sayısal olarak temsil edilmeleri

Yapının Kullanım Şekli	Sayısal Karşılığı
Konut	1
Üretim Tesisi	2
Hastane	3
İbadethane	4
Okul	5
Kütüphane	6
Tehlikeli Madde Deposu	7
Nükleer Tesis	8

Yapının stabilitesi ve hasar durumu

Çizelge 4.1’de yapının durumu stabil ve hasarlı olarak ikiye ayrılmıştır. Bu iki kategori için de Çizelge 4.4’ de sayısallaştırma işlemi yapılmıştır. Hasar durumu da kendi içinde, ağır, orta ve az olmak üzere üç kategoriye ayrılmıştır. Çizelge 4.5’ de yapıların hasar durumlarının sayısallaştırılması işlemi gözükmektedir.

Çizelge 4.4. Yapının stabiltesi ve hasar durumu sayıllaştırılması

Yapının Stabilitesi ve Hasar Durumu	Sayısal Karşılığı
Stabil	0
Hasarlı	1

Çizelge 4.5. Yapının hasar durumu sayıllaştırılması

Yapının Stabilitesi ve Hasar Durumu	Sayısal Karşılığı
Az	1
Orta	2
Ağır	3

Yapının komşu yapılara olan uzaklığı

Bu veri metre cinsinden sürekli veri olması nedeniyle, herhangi bir sayısallaştırma işlemi yapılmamıştır.

Yakınında hastane, okul, ibadethane gibi çok sayıda insanın kullandığı yapıların varlığı ve uzaklıkları

Yıkılacak olan yapının yakınında çok sayıda insanın aynı anda kullandığı bir yapı varsa 1(bir), yoksa 0 (sıfır) olarak sayısallaştırıp, bu yapıların uzaklıkları da metre cinsinden sürekli veri olarak girilecektir.

Yıkılacak yapının altyapı durumu

Yıkılacak yapının altyapısında su, doğalgaz, metro hattı gibi yıkım sırasında zarar görebilecek alt yapılar var ise 1(bir), yoksa 0 (sıfır) olarak sayısallaştırılıp modele eklenecektir.

İzin verilen gürültü seviyesi

Yıkım sırasında ortamda izin verilen gürültü seviyesi dBA cinsinden girilecektir.

İzin verilen titreşim seviyesi

Yıkım sırasında ortamda izin verilen titreşim seviyesi PGV:mm/sn cinsinden girilecektir.

Toz seviyesi

Yıkım sırasında oluşacak toz seviyesi düşük, orta ve yüksek olarak girilecektir. Bu düzeylerin sayıllaştırılması, Çizelge 4.6’ da gösterilmiştir.

Çizelge 4.6. Toz seviyelerinin sayısallaştırılması

İzin Verilen Toz Seviyesi	Sayısal Karşılığı
Düşük	1
Orta	2
Yüksek	3

Yıkım teknikleri

Yıkım tekniklerinin belirlenmesi için uzmanlarla görüşmeler yapılmıştır. Bu görüşmeler sonucunda aşağıdaki yıkım tekniklerinin, bir yapının yıkımında genel olarak uygulandığı görüşüne varılmıştır. Geliştirilen yazılım ise bu tekniklerden birisini, girilen binanın özelliklerine göre önermektedir.

Çizelge 4.7. Yıkım tekniklerinin sayısallaştırılması

Yıkım Tekniği	Sayısal Karşılığı
El ile yıkım	0
30 tonluk ekskavatör ile yıkım	1
Makine ile yıkım	2
Kat eksiltme tekniği ile yıkım	3
Uzun erişimli yıkım makinesi ile yıkım	4
Patlayıcı ile yıkım	5
Önce kat eksiltme, devamında uzun erişimli makine ile yıkım	6
Önce kat eksiltme, devamında makine ya da 30 tonluk ekskavatör ile yıkım	7

4.3.2. Sentetik veri seti oluşturma süreci

Yukarıda açıklandığı üzere, önceden yapılmış yıkımlara ilişkin teknik seçimlerinde kullanılan verilere ulaşım hedefinde yapılan araştırmalar ve taramalar neticesinde bazı gerçek verilere ulaşılmış ancak yapay zeka çalışmalarına uygun veri seti elde edilecek düzeyde olmadıkları tespit edilmiştir.

Bu bölümde kriterlere veri atamaları yaparak 10000 satırlık sentetik bir veri seti oluşturulmuştur. Pratikte, bu veri setinin yıkım uzmanları tarafından girilen verilerle desteklenmesi planlanmaktadır. Veri seti oluşturulurken python programlama dili

kullanılmış olup, makine öğrenmesi algoritmaları tarafından kullanılabilmeleri amacıyla .csv uzantılı bir dosyada saklanmıştır. (Csv- Comma-Separated Values). Veri setinin oluşturulması amacıyla yazılan kodlar Şekil 4.1‘ de gösterilmiştir.

```
import numpy as np
import pandas as pd
ornekSayisi = 10000
np.random.RandomState(0)
def veriuret(aralikMin,aralikMax,size=1):
    return np.random.randint(aralikMin,aralikMax,size)
def veriuretWStep(aralikMin,aralikMax,size=1,step=0):
    if step:
        liste = [i for i in range(aralikMin,aralikMax+1,step)]
        return np.random.choice(liste,size)

liste = ["yukseklik","tasiyici","kullanim","durum",\
         "hasar","komsu","kamubina","sudogalgaz","gurultu","toz","titresim
","yolmesafe","sonuc"]
sozluk = {}
sozluk[liste[0]] = veriuretWStep(3,60,ornekSayisi,3)  #yukseklik
sozluk[liste[1]] = veriuret(1,6,ornekSayisi) #tasiyici
sozluk[liste[2]] = veriuret(1,7,ornekSayisi) #kullanim
sozluk[liste[3]] = veriuret(0,2,ornekSayisi) #Durum
sozluk[liste[4]] = veriuret(1,4,ornekSayisi)  #Hasar
sozluk[liste[5]] = veriuretWStep(5,100,ornekSayisi,5) #Komsu
sozluk[liste[6]] = veriuret(0,2,ornekSayisi) #kamubina
sozluk[liste[7]] = veriuret(0,2,ornekSayisi) #sudogalgaz
sozluk[liste[8]] = veriuret(20,81,ornekSayisi) #gurultu
sozluk[liste[9]] = veriuret(1,5,ornekSayisi) #toz
sozluk[liste[10]] = veriuret(200,601,ornekSayisi) #titresim
sozluk[liste[11]] = veriuretWStep(5,100,ornekSayisi,5) # yol trafik mesafe
sozluk[liste[12]] = veriuret(0,8,ornekSayisi) #sonuc
bilgi = pd.DataFrame(sozluk)
#-----------------------------------------------------------------
bilgi.loc[(bilgi.yukseklik<=6),"sonuc"] = 0
bilgi.loc[((bilgi.yukseklik>6)&(bilgi.yukseklik<=12)),"sonuc"] = 1
bilgi.loc[((bilgi.yukseklik>12)&(bilgi.yukseklik<=27)),"sonuc"] = 2
bilgi.loc[((bilgi.yukseklik>27)&(bilgi.yukseklik<=36)),"sonuc"] = 3
bilgi.loc[(bilgi.yukseklik>36)&(bilgi.komsu<50),"sonuc"] = 3
#-----------------------------------------------------------------
bilgi.loc[(bilgi.sudogalgaz==1)&(bilgi.komsu>50)&(bilgi.yukseklik>36)&(bil
gi.sonuc==5),"sonuc"] = 2
bilgi.loc[(bilgi.sonuc==5)&(bilgi.kamubina==1),"sonuc"]=3
bilgi.loc[(bilgi.durum==0)&(bilgi.sonuc==5),"sonuc"] = 2
bilgi.loc[(bilgi.sonuc==5)&(bilgi.titresim>300)&(bilgi.toz>2),"sonuc"]=3
```

Şekil 4.1. Sentetik veri üretimini için yazılan python kodları ekran görüntüsü

Veri setinin geliştirilmesinde, yıkım uzmanlarının görüşleri doğrultusunda üretilen senaryoların uygun yıkım tekniği ile etiketlenmesi yapılmıştır. Bunun amacı, verilerin tutarlı olmasını sağlayarak, makine öğrenmesi modelleri ile eğitim işlemlerindeki doğruluk oranlarını arttırmaktır.

Sentetik veri setinin geliştirilmesi aşamasında, tamamen rastsal değerler yerine, gerek akademik gerekse de sahadaki yıkım uzmanları ile görüşülerek, hem gerçek değerlere uygun, hem de fiili olarak olması pek mümkün olmayan verilerden oluşan veri seti geliştirilmiştir. Bu veri setinin fiili olarak gerçekleşmesi zor olan verileri içermesinin amacı ise, makine öğrenmesi süreçlerinden birisi ve en önemlisi olan eğitim sürecinde farklı verilerin de değerlendirilmesini sağlamaktır.

Veri seti üretimi amacı ile geliştirilen kodların çalıştırılması sonucunda, Şekil 4.2‘ de 10 adet örneği görülen veriler elde edilmiştir.

	yukseklik	tasiyici	kullanim	durum	hasar	komsu	kamubina	sudogalgaz	gurultu	toz	titresim	yolmesafe	sonuc
0	54	3	3	1	1	80	1	1	56	4	559	70	0
1	6	1	5	0	3	70	0	0	59	4	389	50	0
2	18	3	1	1	3	80	0	1	51	3	403	5	2
3	42	2	2	0	2	5	1	0	20	3	431	60	3
4	12	1	6	0	1	20	0	1	54	1	435	55	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
9995	60	3	1	1	2	95	0	0	75	3	224	55	0
9996	15	5	3	0	2	35	1	1	23	4	543	85	2
9997	39	5	2	0	1	65	1	0	70	1	485	15	4
9998	6	5	3	1	2	55	1	0	65	4	558	30	0
9999	33	2	3	1	1	20	0	1	31	4	375	70	3

10000 rows × 13 columns

Şekil 4.2. Oluşturulan veri seti ekran görüntüsü

Veri seti oluşturulurken yıkım uzmanları ile yapılan görüşmeler sonucunda ulaşılan önemelerle, verilerin kararlı hale getirilmesi amaçlanmıştır. Bu çerçevede, yıkım yöntemi kararları için daha efektif sonuçlar elde edilmesi planlanmaktadır. Bu önermeler, veri setinin rastgele verilerle üretilmesinin ardından elde edilen değerler üstünden uygulanmıştır. Önermelerin, veri setine dahil edilmesiyle ilgili python kodları da yine önermelerle birlikte belirtilmiştir.

Önerme 1

Yıkılacak yapı herhangi bir başka yapı ile bitişik nizamda imal edilmişse patlayıcı ile yıkım tekniği kullanılamaz.

```
bilgi.loc[((bilgi.komsu<10)&(bilgi.sonuc==5)),"sonuc"] = 5
```

Önerme 2

Yıkılacak yapının 50 metre yakınında metro, ibadethane, kütüphane, okul, hastane, güç istasyonu, doğalgaz hattı vb. tesisler bulunuyorsa patlayıcı ile yıkım tekniği kullanılamaz.

```
bilgi.loc[((bilgi.komsu<=50)&(bilgi.sudogalgaz == 1 | bilgi.kamubina))
,"sonuc"] = 5
```

Önerme 3

Yıkılacak yapı yüksekliğinin yarısından (h/2) daha yakında başka yapı, yaya kaldırımı, araç trafiğinin mevcut olduğu yol bulunuyorsa ve yapı yüksekliği 27 metre (9 katlı) ve daha alçak ise 12 metre yüksekliğe kadar uzun erişimli yıkım makinesi kullanılmalıdır.

```
bilgi.loc[(((bilgi.yukseklik//2>=bilgi.komsu)|(bilgi.yukseklik//2>=
bilgi.yolmesafe))&(bilgi.yukseklik<=27)&(bilgi.yukseklik>12)),"sonu
c"] = 4
```

Önerme 4

Yıkılacak yapı yüksekliğinin yarısından (h/2) daha yakında başka yapı, yaya kaldırımı, araç trafiğinin mevcut olduğu yol bulunuyorsa ve yapı yüksekliği 27 metre (9 katlı) ve daha yüksek ise öncelikle 27 metreye kadar kat eksiltme tekniği kullanılarak yıkıma başlanır. Daha sonra uzun erişimli yıkım makinesi kullanılmalıdır.

```
bilgi.loc[(((bilgi.yukseklik//2>=bilgi.komsu)|(bilgi.yukseklik//2>=
bilgi.yolmesafe))&(bilgi.yukseklik>27)),"sonuc"] = 6
```

Önerme 5

Yıkılacak yapı yüksekliğinin yarısından (h/2) daha yakında başka yapı, yaya kaldırımı, araç trafiğinin mevcut olduğu yol bulunuyorsa ve yapı yüksekliği 27 metre (9 katlı) ve daha yüksek ise öncelikle 27 metreye kadar kat eksiltme tekniği kullanılarak yıkıma başlanır. Kat döşemesi statik durumu mini makine kullanımına olanak vermiyorsa ya da mini makineyi bina üzerine çıkartmak için gerekli vinç kurulum alanı yoksa kat eksiltme el ile yıkım tekniği kullanılarak yapılır.

```
bilgi.loc[(((bilgi.yukseklik//2>=bilgi.komsu)|(bilgi.yukseklik//2>=
bilgi.yolmesafe))&(bilgi.yukseklik>27)&(bilgi.durum==1)),"sonuc"] =
 7
```

Önerme 6

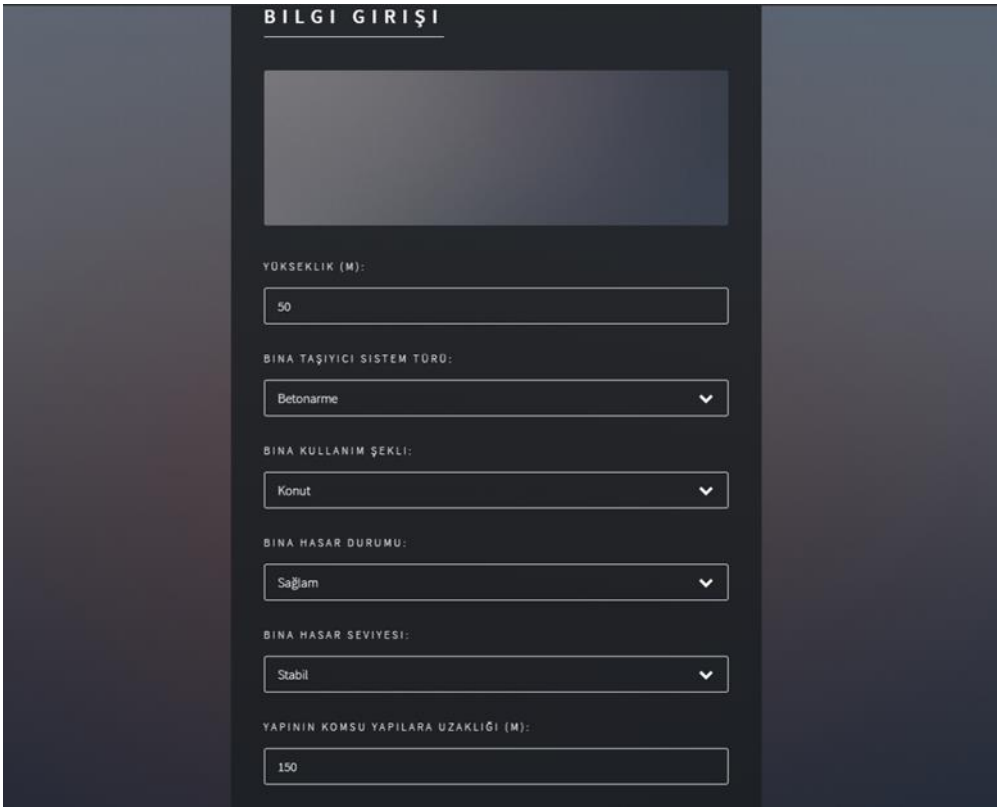
Yıkılacak yapı yüksekliğinin yarısından (h/2) daha yakında başka yapı, yaya kaldırımı, araç trafiğinin mevcut olduğu yol bulunuyorsa ve yapı yüksekliği 12 metre (4 katlı) ve daha alçak ise standart 30 tonluk ekskavatör kullanılarak yapı yıkımı yapılır.

```
bilgi.loc[(((bilgi.yukseklik//2>=bilgi.komsu)|(bilgi.yukseklik//2>=
bilgi.yolmesafe))&(bilgi.yukseklik<=12)),"sonuc"] = 1
```

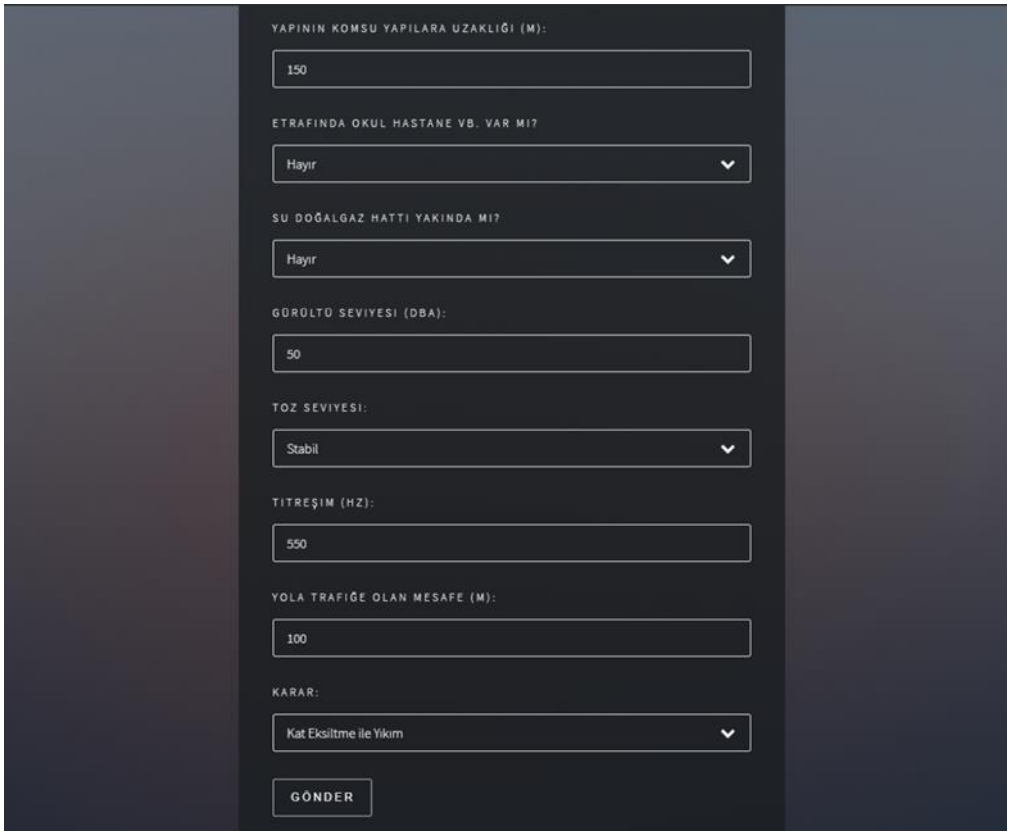
Üretilen veriler, uzman görüşleri ile revize edildikten sonra, verilerin özelliklerine uygun makine öğrenmesi algoritmaları belirlenerek, bu veriler arasındaki ilişkilerin istatistiksel olarak ortaya çıkarılması sağlanmıştır. Bu aşamada makine öğrenmesi modellerinden faydalanılmıştır.

Eğitim sonrasında sisteme dışarıdan veri girilmesi ile denemeler yapılmış, girilen bu veriler aynı zamanda uzman görüşüne sunularak, sistemin verdiği karar ile uzmanların verdiği kararların örtüşme seviyelerine bakılmıştır. Sistem ve uzmanların verdiği kararlar arasında farklılıklar görüldüğünde, sistemin eğitimi revize edilerek örtüşen sonuçlar elde edilmeye çalışılmıştır. Bu süreç örtüşen sonuçlar elde edilinceye kadar iteratif biçimde işletilmiştir. Sürecin tamamlanması ile ikinci aşamaya geçilmiş ve verilerin reel veriler ile nasıl destekleneceği konusunda web tabanlı bir sistem geliştirilmiştir.

İkinci aşamada, reel verilerin toplanması amacıyla, bir ara yüz geliştirilmiş ve web üzerinden yayınlanmıştır. Bu ara yüzde, ülkemizdeki regülasyonlarla belirlenen yıkım teknikleri seçiminde kullanılan değişkenlerin, kullanıcılar tarafından girilerek sisteme aktarılması ve sistemin daha önce işletilen uzman görüşü destekli ve makine öğrenmesi süreçlerinden geçen verilerden yola çıkarak bir tahmin yapması sağlanmıştır. Bu aşamada sistemin tahmin doğruluğu önceki eğitim verileri ile sınırlı olduğundan, karar destek amacıyla kullanılabilir durumdadır. Geliştirilen web tabanlı ara yüz bölüm 5.3.3'de detaylı olarak ele alınmıştır.



Şekil 4.3. Verilerin girişi amacıyla geliştirilen ara yüz

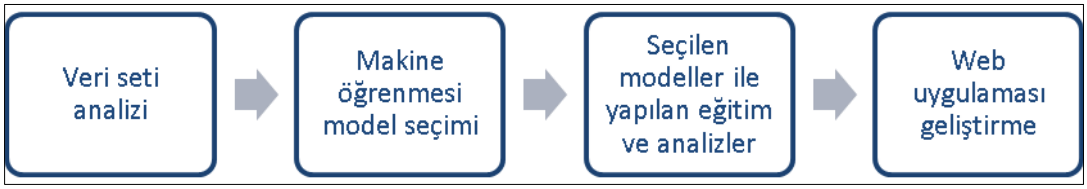


Şekil 4.3. (devam)Verilerin girişi amacıyla geliştirilen ara yüz

Sistemin tahmin kararlılığını arttırmak amacıyla, web üzerindeki arayüz ile uzmanların tekrarlı veri girmesi ve optimal değerlendirmelere ilişkin kararlarını işaretleyebileceği bir web ortamı da ayrıca oluşturulmuştur. Bu şeklide oluşturulan makine öğrenmesi sistemi kararların stabilitesi ve doğruluğu üzerinden verimliliğinin artırılması hedeflenmiş geçerliliği uzman görüşü karşılaştırmaları ile kontrol edilmiş başarılı yıkım proje sonuçlarının yer aldığı bir veri seti de oluşturulmuştur.

5. BULGULAR

Bu bölümde, Şekil 5.1'de sunulan sıra ile veri setinin analizi ve eğitim için kullanılan makine öğrenmesi modellerinden elde edilen çıktılara ve yazılımın eğitimine dair karar alma süreçlerine ilişkin bulgulara yer verilmiştir. Diğer taraftan elde edilen sonuçlar ile birlikte oluşturulan web tabanlı uygulama detayları bölüm 5.3'de sunulacaktır.



Şekil 5.1. Bulgular bölümü içerik şeması

5.1. Üretilen Sentetik Veri Setinin Analizi

Bölüm 4.1.2'de yapılan araştırmalar ve taramalar neticesinde ulaşılan verilerin yapay zeka çalışmalarına uygun seviyeye getirilebilmesi amacıyla 10000 satırlık sentetik bir veri seti oluşturulmuştur. Bu aşamada python programlama dili kullanılmış olup, makine öğrenmesi algoritmaları tarafından kullanılabilmeleri amacıyla .csv uzantılı bir dosyada saklanmıştır. Oluşturulan kodlar yöntem bölümünde Şekil 4.1' de verilmiştir. Veri seti hazırlığının arkasından ilgili analizler yapılmıştır. Bu kapsamda;

- Sonuçların veri setindeki dağılımı,
- Yıkım tekniğinde kullanılan kriterlere gelen değerlerin dağılımı,
- Sonuçların sayıları,
- Verilerin tiplerinin belirlenmesi,
- Satırlar içerisinde değersiz kriter olup olmadığı,
- Verilerin sayısı, ortalaması, standart sapması, minimum ve maksimum değerleri, %25'lik, %50'lik, %75'lik değerleri,
- Verilerin sonuca etki oranları,

incelenmiştir. Bu analizlerin gerçekleştirilebilmesi için Şekil 5.2.' de belirtilen, python için geliştirilmiş kütüphanelerden faydalanılmıştır.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.svm import LinearSVC
from sklearn.tree import DecisionTreeRegressor
from sklearn.preprocessing import MinMaxScaler as Scaler
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.model_selection import RandomizedSearchCV,GridSearchCV
from sklearn.metrics import confusion_matrix,classification_report
from sklearn.metrics import precision_score,recall_score,f1_score
from sklearn.metrics import roc_curve
```

Şekil 5.2. Kullanılan makine öğrenmesi kütüphaneleri ekran görüntüsü

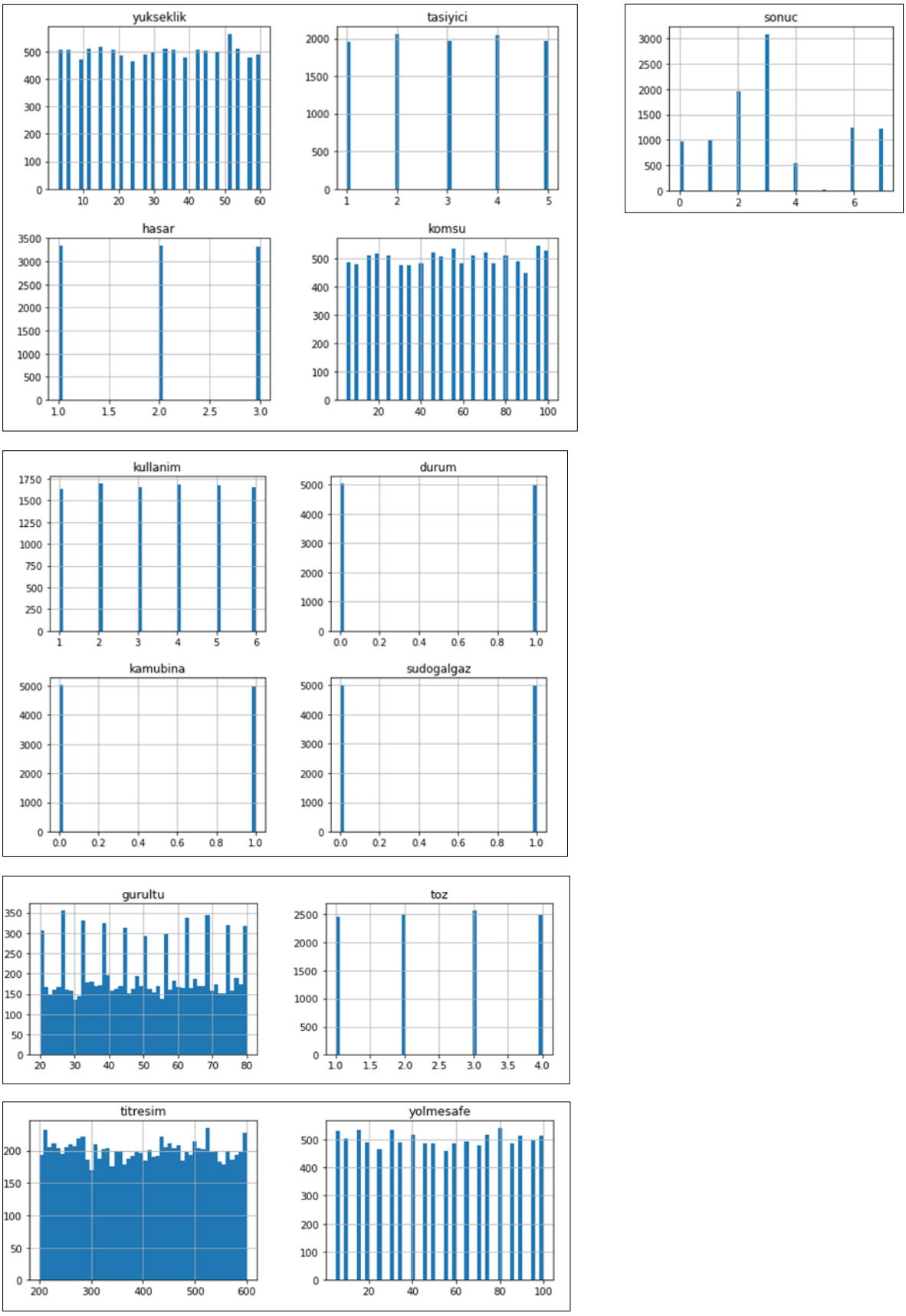
Sonuçların veri setindeki dağılımı

Bir yıkım projesinde kullanılabilecek yıkım tekniklerinin, üretilen sentetik veri setindeki oransal dağılımı Şekil 5.3'deki gibi gerçekleşmiştir.

```
liste = [0,1,2,3,4,6,7]
bilgi.loc[(bilgi.sonuc<0),"sonuc"] = np.random.choice(liste)
bilgi.sonuc.value_counts(normalize=True)
3    0.3076
2    0.1954
6    0.1233
7    0.1216
1    0.0991
0    0.0965
4    0.0546
5    0.0019
Name: sonuc, dtype: float64
```

Şekil 5.3. Sentetik veri setindeki sonuçların oransal dağılımları ekran görüntüsü

Veri setindeki sonuç etiketlerinin oransal dağılımları incelendiğinde 3 değeri ile etiketlenmiş olan kat eksiltme ile yıkım tekniğinin en yüksek orana sahip olduğu görülmüştür.
Şekil 5.3'deki oransal dağılımların veri setindeki sayılarının grafikleri çıkarılmıştır.



Şekil 5.4. Veri setinde bulunan kriterlere gelen değerlerin sayılarının grafiksel gösterimi

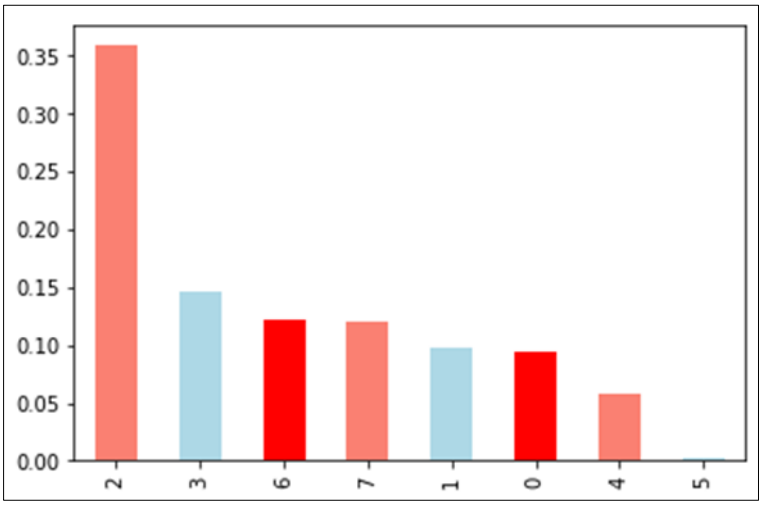
Sonuçların sayısal olarak dağılımları, grafiksel incelemeden sonra rakamsal olarak da değerlendirilmiştir. Bunun için Şekil 5.5'de ekran görüntüsü gösterilen kod yazılmıştır.

```
bilgi["sonuc"].value_counts()


3    3076
2    1954
6    1233
7    1216
1     991
0     965
4     546
5      19
Name: sonuc, dtype: int64
```

Şekil 5.5. Veri setindeki sonuç değerlerinin sayıları ekran görüntüsü

Veri setindeki sonuç değerlerinin, tüm değerlere oranları da grafiksel olarak incelenmiştir. Şekil 5.6' da sonuç olarak etiketlenmiş verilerin tüm veri setindeki oransal dağılımlarının grafiksel olarak gösterilmesi için yazılan kodlar ve elde edilen grafik görülmektedir.

```
bilgi["sonuc"].value_counts(normalize=True).plot(kind="bar",color=["
salmon","lightblue","red"])
```



Şekil 5.6. Veri setindeki sonuç değerlerinin oransal dağılımları grafiği ve yazılan kodlar ekran görüntüsü

Veri setindeki verilerin tipleri

Veri setindeki kategorik veriler sayıllaştırılarak ifade edilmiştir. Bu anlamda elde edilen sayısal verilerin tipleri info() fonksiyonu ile görüntülenmiştir. Tüm verilerin sayısal olması nedeniyle 64 bitlik tam sayı olan int64 veri tipinde olduğu gözlemlenmiştir. Veri tiplerini görüntülemek için kullanılan info() fonksiyonu ile aynı zamanda verilerin sayıları da elde edilmiştir. Çizelge 5.1' da gösterilen sonuçlar incelendiğinde her bir veriden on bin adet olduğu görülmektedir.

Çizelge 5.1. Veri setindeki verilerin tipleri ve sayıları

```
<class 'pandas.core.frame.DataFrame'>

RangeIndex: 10000 entries, 0 to 9999
Data columns (total 13 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   yukseklik   10000 non-null  int64
 1   tasiyici    10000 non-null  int64
 2   kullanim    10000 non-null  int64
 3   durum       10000 non-null  int64
 4   hasar       10000 non-null  int64
 5   komsu       10000 non-null  int64
 6   kamubina    10000 non-null  int64
 7   sudogalgaz  10000 non-null  int64
 8   gurultu     10000 non-null  int64
 9   toz         10000 non-null  int64
 10  titresim    10000 non-null  int64
 11  yolmesafe   10000 non-null  int64
 12  sonuc       10000 non-null  int64
dtypes: int64(13)
memory usage: 1015.8 KB
```

Üretilen veriler içerisinde boş değer kontrolü

Bu kontrol ile, üretilen satılarda boş değer olup olmadığına bakılmaktadır. Sonuç olarak 0 çıkması, boş değer olmadığını göstermektedir. Veri setinin regülasyonlara ve uzman görüşlerine dayalı olarak sentetik üretilmesi nedeniyle boş değer bulunmamaktadır. Verilerde boş değerlerin olmaması, makine öğrenmesi algoritmalarının veriler arasındaki ilişkileri hesaplarken kullandığı yöntemlerde etkili sonuçlar elde edilmesi

sağlanabilmektedir. Boş değer kontrolü için “bilgi.isna.sum()” fonksiyonundan faydalanılmıştır. Elde edilen sonuçlar Çizelge 5.10’ da gösterilmiştir.

Çizelge 5.2. Üretilen veri seti içerisinde yapılan boş değer kontrolü sonuçları

yukseklik	0
tasiyici	0
kullanim	0
durum	0
hasar	0
komsu	0
kamubina	0
sudogalgaz	0
gurultu	0
toz	0
titresim	0
yolmesafe	0
sonuc	0

Verilerin tipleri, sayıları ve boş veri kontrolleri ardından, her bir kriter için üretilen değerlerin tüm veri seti içindeki sayısı, ortalaması, standart sapması, minimum ve maksimum değerleri, %25’lik, %50’lik, %75’lik değerlerini görmek amacıyla describe() fonksiyonundan faydalanılmıştır. Verilerin istatistikleri Çizelge 5.3’ de gösterilmiştir.

Çizelge 5.3. Veri setindeki veriler ile ilgili istatistikler

	yukseklik	tasiyici	kullanim	durum	hasar	komsu	kamubina	sudogalgaz	gurultu	toz	titresim	yolmesafe	sonuc
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	31.573800	3.000900	3.505400	0.497000	1.998900	52.780500	0.495900	0.498500	50.123300	2.508400	398.834200	52.412000	3.231600
std	17.324871	1.408012	1.702193	0.500016	0.815699	28.830443	0.500008	0.500023	17.592869	1.112049	116.345259	29.084908	2.127114
min	3.000000	1.000000	1.000000	0.000000	1.000000	5.000000	0.000000	0.000000	20.000000	1.000000	200.000000	5.000000	0.000000
25%	15.000000	2.000000	2.000000	0.000000	1.000000	30.000000	0.000000	0.000000	35.000000	2.000000	296.000000	25.000000	2.000000
50%	33.000000	3.000000	4.000000	0.000000	2.000000	55.000000	0.000000	0.000000	50.000000	3.000000	400.000000	50.000000	3.000000
75%	48.000000	4.000000	5.000000	1.000000	3.000000	80.000000	1.000000	1.000000	65.000000	3.000000	500.000000	80.000000	4.000000
max	60.000000	5.000000	6.000000	1.000000	3.000000	100.000000	1.000000	1.000000	80.000000	4.000000	600.000000	100.000000	7.000000

Verilerin sonuca etki oranları

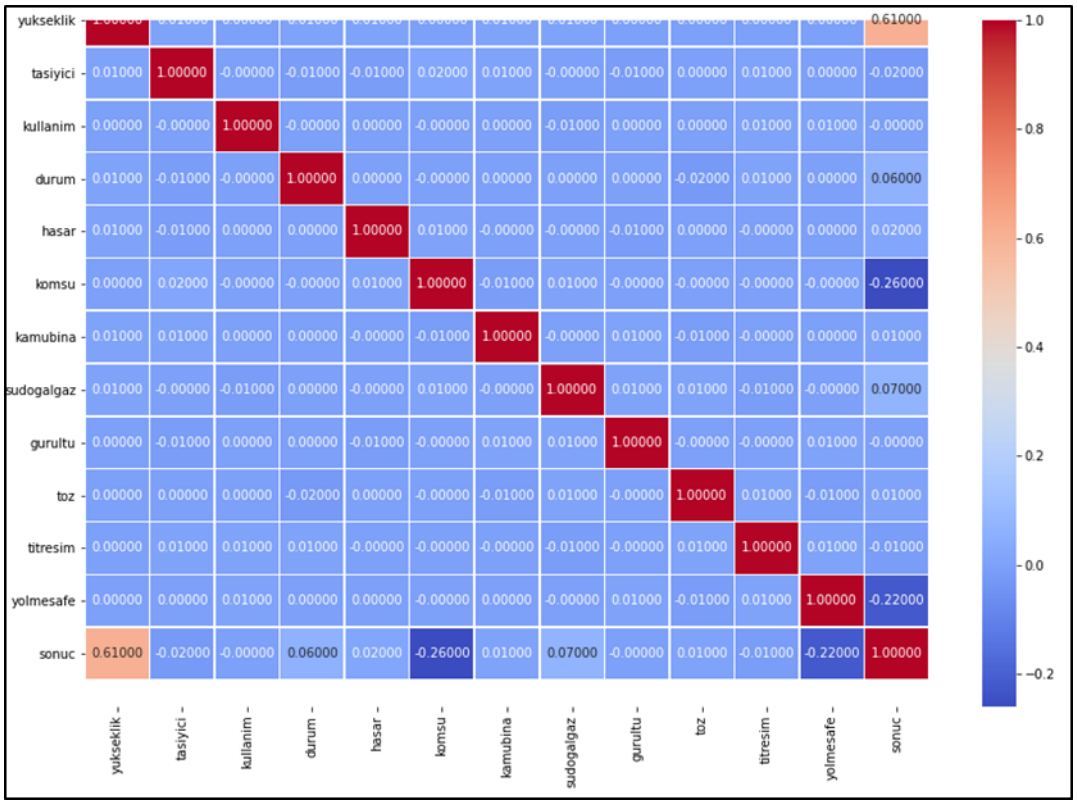
Verilerin sonuca etki oranlarını gözlemleyebilmek için korelasyon matrisi oluşturulmuştur. Bu matris ile veri setindeki hangi verinin sonuca daha fazla etki ettiği gözlemlenebilmektedir. Korelasyon matrisi oluşturmak amacıyla yazılan kodların ekran görüntüsü Şekil 5.7'de, görülmektedir.

```
cor_matrix =  bilgi.corr().round(2)
fig,ax = plt.subplots(figsize=(15,10))
sns.heatmap(cor_matrix,cmap='coolwarm', annot=True, linewidths=.5,fmt=
".5f", ax=ax)
bottom,top = ax.get_ylim()
ax.set_ylim(bottom+0.5,top=0.5)
```

Şekil 5.7. Verilerin sonuca etki oranları korelasyon matrisi oluşturmak için yazılan kodların ekran görüntüsü

Şekil 5.7' de gösterilen kodlar sonucunda elde edilen korelasyon matrisi ısı haritası Çizelge 5.4' de gösterilmiştir.

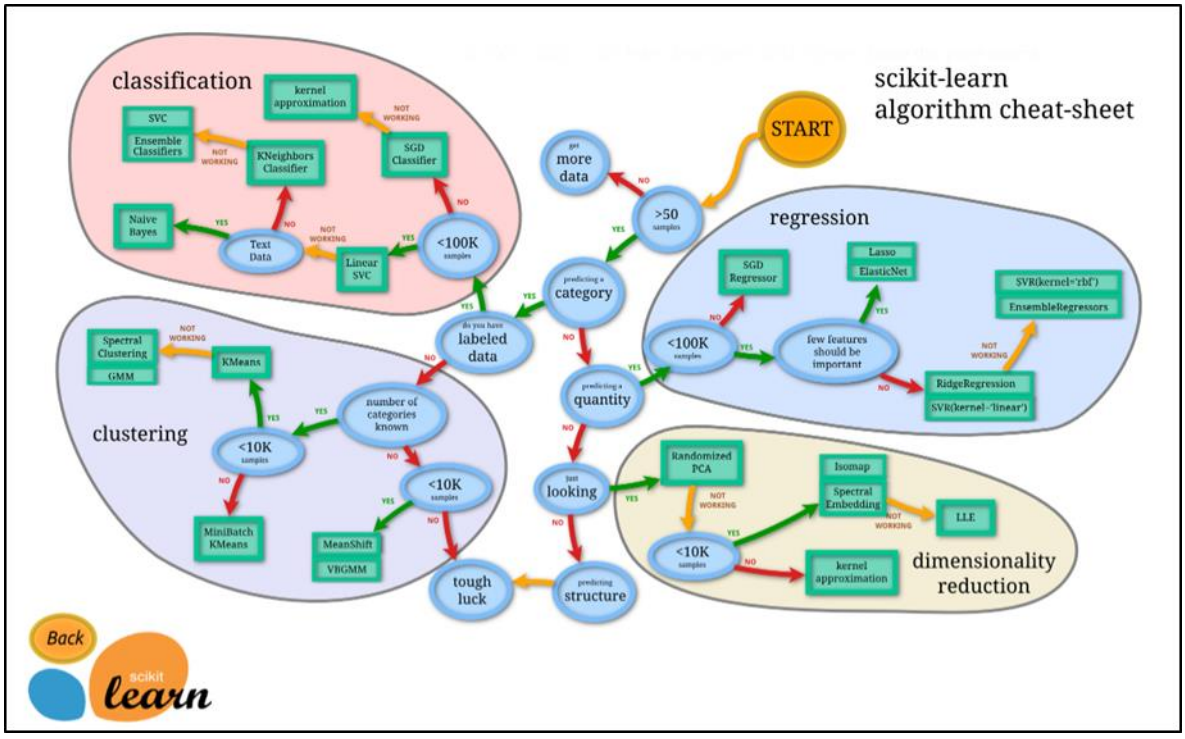
Çizelge 5.4. Verilerin sonuca etki oranları korelasyon matrisi ısı haritası



Çizelge 5.4'de gösterilen korelasyon matrisi ısı haritasına göre, veri setindeki veriler temel alındığında yıkım tekniği seçiminde en önemli faktörün yapı yüksekliği olduğu gözlemlenmektedir. Bu durum uzman görüşleri ve literatür ile uyumlu olup veri setinin tutarlılığı açısından önemli bulunmuştur.

5.2. Makine Öğrenmesi Model Seçimi

Makine öğrenmesi problemlerinde doğru makine öğrenmesi modelinin seçimi çok önemlidir. Bu aşama genelde en zor aşamadır. Veri setinde bulunan verilerin özellikleri, veri setinin büyüklüğü, etiketlenmiş verinin bulunması vb. durumlar doğru makine öğrenme modelinin seçiminde rol oynamaktadır (Pedregosa ve diğerleri, 2011). Eldeki veri setine uygun makine öğrenmesi modelinin seçimi için tez süresince kullanılmış olan scikit learn makine öğrenme kütüphanesi tarafından sunulan akış şeması dikkate alınmıştır ve Şekil 5.8'de gösterilmiştir.



Şekil 5.8. Makine öğrenmesinde veri setine uygun makine öğrenmesi modeli seçimi

Pedregosa ve arkadaşlarının geliştirdiği Şekil 5.8'de gösterilen makine öğrenmesi modeli seçimi akış şeması incelendiğinde, üretilen sentetik veri setinin özellikleri ile ilerlenmeye başlandığında elde edilenler aşağıdaki gibidir;

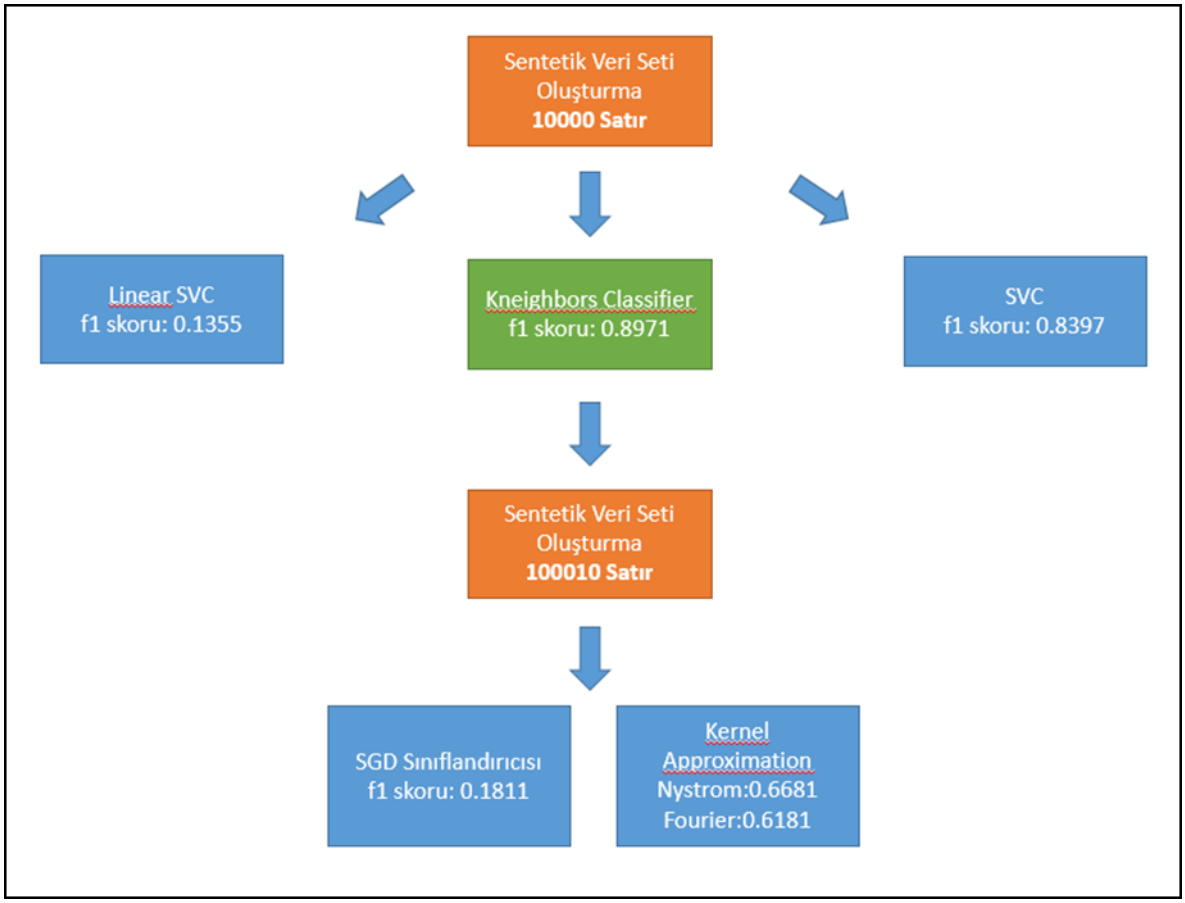
- Veri seti 50 örnekten fazladır
- Kategorik tahmin yapılmaktadır
- Veri setindeki veriler etiketlenmiştir
- Veri setindeki veriler 100000 veriden azdır

Yukarıdaki yol izlendiğinde, yıkım tekniklerinin seçiminde sınıflandırma modellerinden olan Linear Svc kullanılması gerektiği görülmektedir. Linear Svc sınıflandırıcısı ile elde edilen f1 skoru değeri 0.1355 olarak elde edilmiştir ve oldukça düşüktür. Linear Svc'nin beklenen sonucu vermemesi nedeniyle yol haritasında devam edilmiştir. Veri setindeki veriler metinsel veriler olmadığından Navie Bayes sınıflandırıcısı yolu izlenmemiştir. Bunun yerine k en yakın komşu sınıflandırıcısı ile denemeler yapılmıştır.

K en yakın komşu sınıflandırıcısı ile yapılan denemelerde elde edilen f1 skoru değeri 0.8971 olarak ölçülerek yüksek bir değer elde edilmiştir

Her ne kadar k en yakın komşu sınıflandırıcısı ile elde edilen değerler yüksek de olsa, akış şemasındaki yola göre, k en yakın komşu sınıflandırıcısından sonra gelen SVC'de denenmiştir. SVC ile alınan f1 skoru değeri, 0.8397 olarak elde edilmiştir.

Bu noktaya kadar yapılan denemelerde, veri setindeki veri sayısı on bin satır olarak belirlenmiştir. Şekil 5.8'deki akış şeması incelendiğinde, yüz bin veriden fazla veri olduğunda, SGD Sınıflandırıcısı ve kernel approximation tekniklerinin önerildiği görülmektedir. Bu çerçeveden bakıldığında, bu tekniklerin de denenmesi amacıyla, veri seti tekrar yüz bin on veri ile yeniden üretilerek denemeler yapılmıştır. SGD sınıflandırıcısı 0.1811 f1 skorunu elde edebilmiştir. Kernel approximation ise, nystroem yöntemi ile 0.6681, fourier yöntemi ile 0.6181 başarım skorunu elde etmiştir. Mevcut duruma göre, yıkım teknikleri seçiminde kullanılacak efektif makine öğrenmesi modelinin k en yakın komşu sınıflandırıcısı olduğu görülmektedir.



Şekil 5.9. Üretilen veri seti için uygun makine öğrenmesi model seçimi

Üretilen sentetik veri setindeki verilerin eğitimi için python programlama dili kullanılarak, seçilen makine öğrenmesi modelleri ve yöntemleri ile eğitim süreçleri işletilmiştir. Eğitim süreçleri sonucunda her bir model ve yöntem için elde edilen sonuçlar karşılaştırılarak, en iyi performansı gösteren model ile eğitim süreci tamamlanıp, web uygulamasında kullanılmak üzere kaydedilmiştir. Bu bölümde, makine öğrenmesi modelleri ve yöntemlerinin kullanımı amacıyla yazılan python kodları ve sonuçları açıklanmıştır. Seçilen model ve yöntemler;

On bin satırlık veri için kullanılan sınıflandırma algoritmaları;

- Doğrusal destek vektör makineleri (Linear support vector machines – Linear svc),
- K en yakın komşu algoritması (Kneighbors algoritması),
- Destek vektör sınıflandırıcı (Support vector classifier – Svc)

Yüz bin on satırlık veri için kullanılan başarım optimizasyon teknikleri;

- Olasılıksal dereceli azalma yöntemi (Stochastic gradient descent- Sgd),

- Çekirdek yaklaşımı yöntemi (Kernel approximation)

5.2.1. Doğrusal destek vektör sınıflandırıcısı (Linear support vector classifer-Linear svc) makine öğrenmesi modeli için geliştirilen kodlar ve sonuçları

Linear svc, svc'nin kernel parametresinin lineer değerine sahip olan bir versiyonudur. Doğrusal vektör ile sınıflandırma yapması sebebi ile sınıflandırma işlemlerinde çok sayıda örnekte iyi sonuçlar elde edilebilmektedir.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

from sklearn.svm import LinearSVC
from sklearn.preprocessing import MinMaxScaler as Scaler
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.model_selection import RandomizedSearchCV,GridSearchCV
from sklearn.metrics import confusion_matrix,classification_report
from sklearn.metrics import precision_score,recall_score,f1_score,accurac
y_score
from sklearn.metrics import roc_curve,plot_confusion_matrix,plot_roc_curv
e

bilgi = pd.read_csv("data.csv")
```

Şekil 5.10. Linear svc modeli kullanımı için kullanılan kütüphaneler ekran görüntüsü

Şekil 5.10' da Linear svc modelinin kullanımı için gerekli olan, python programlama dili için geliştirilmiş kütüphaneler görülmektedir. Kodun son satırında, üretilen ve csv formatında kaydedilen on bin satırlık veri setinin okuma işlemi gerçekleştirilmiştir.

```
X = bilgi.drop("sonuc",axis=1)
y = bilgi["sonuc"]
X_egit,X_test,y_egit,y_test = train_test_split(X,y,test_size=0.2)
scaler = Scaler()
scaler.fit(X_egit)
train_set_scaled = scaler.transform(X_egit)
test_set_scaled = scaler.transform(X_test)
scaled_X = scaler.transform(X)
pd.DataFrame(scaled_X).head()
```

Şekil 5.11. Veri setinin test ve eğitim verisi olarak bölünürken yapılan ölçeklendirme işlemi ekran görüntüsü

Şekil 5.11' de sonuç olarak isimlendirilen ve yıkım tekniklerinin sayısal olarak ifade edildiği sütun, veri setinden çıkarılarak farklı bir değişkene aktarılmıştır. Veriler arasındaki aralık farklılıklarını optimize etmek amacıyla ölçeklendirme işlemi yapılarak, veriler her makine öğrenmesi modeli oluşturma aşaması öncesinde olduğu gibi eğitim-test verileri olarak ayrıştırılmıştır.

Bu noktada "test_size = 0.2" değeri ile veri setindeki verilerin yüzde yirmilik bir kısmı test verisi olarak ayrılmıştır. Ayrıştırılacak verilerin seçimi rastgele olarak gerçekleştirilmektedir. "Train_test_split" metodu üzerinden gerçekleştirilen bu işlem sırasında isteğe bağlı olarak rastlantısal durumun, "random_state" parametresine değer girilmesi yoluyla sabitleme işlemi gerçekleştirilebilir. Bu işlemin ardından "X_egit" değişkeninde barındırılan veriler makine öğrenme modelini eğitmek için "X_test "değişkeninde barındırılan veriler ise bu eğitim sürecinin başarımını değerlendirmek için kullanılacaktır.

Eğitim ve test verisi olarak ayrıştırma işleminin ardından; sentetik üretilmiş olan veri seti içerisindeki ayrık verilerden arındırmak (normalizasyon) bir başka ifadeyle öğrenme sürecini aksatmaya uğratacak farklılıkları gidermek ve bazı makine öğrenme modellerinin eğitim süresini ve başarımın olumlu yönde etkilemek amacıyla ölçeklendirme işlemi yapılmaktadır.

Akış şeması doğrultusunda tercih edilen makine öğrenme modellerinin çoğunun uzaklık temelli olması bir başka ifadeyle veri setinde yer alan değişim aralıklarına karşı daha hassas

olması nedeniyle ölçeklendirme işlemi "MinMaxScaler" metodu ile gerçekleştirilmiştir. Bu sayede aykırı değerlerin etkisini bastırabilecek Şekilde standart sapmalarla sonuçlanan değerler elde edilmektedir.

$$X_{yeni} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (5.1)$$

Ölçeklendirme işlemi sonrasında veri setindeki ilk beş satırın ekran görüntüsü Çizelge 5.13' de gösterilmiştir.

Çizelge 5.5. Ölçeklendirme işlemi sonrası veri setindeki ilk beş satır

	0	1	2	3	4	5	6	7	8	9	10	11
0	0.736842	1.00	0.2	1.0	0.0	0.578947	0.0	1.0	0.266667	1.000000	0.9200	0.789474
1	0.789474	0.75	1.0	1.0	0.5	0.105263	1.0	0.0	0.116667	0.000000	0.7575	0.526316
2	0.157895	0.00	0.2	1.0	1.0	0.000000	0.0	1.0	0.250000	1.000000	0.8200	0.578947
3	0.842105	1.00	0.0	0.0	0.0	1.000000	1.0	1.0	0.916667	0.333333	0.4850	0.210526
4	0.684211	0.50	0.6	0.0	0.5	0.789474	1.0	1.0	0.416667	0.666667	0.9025	0.315789

Verilerin ölçeklendirme işlemi sonrasında k katmanlı çapraz doğrulama işlemi ölçeklenmiş veriler üzerinde çalıştırılarak, birden fazla eğitim-test grubunda modelin performansına yönelik daha fazla bilgi elde edilmiştir. Şekil 5.12'de yazılan kodlar görülmektedir.

```
 svc = LinearSVC(multi_class="crammer_singer")
print(*svc.get_params(),sep="\n")
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
kfold = KFold(n_splits=40)
sonuc = cross_val_score(svc, scaled_X, y, cv=kfold, scoring='accuracy
')
print(sonuc.mean())

C
class_weight
dual
fit_intercept
intercept_scaling
loss
max_iter
multi_class
penalty
random_state
tol
verbose
0.46609999999999996
```

Şekil 5.12. K katmanlı çapraz doğrulama için yazılan kodların ekran görüntüsü

Veri seti "train_test_split" metodu ile ayrıştırılmıştır. Bu durumda ayrıştırma sırasında her ne kadar rastlantısal bir seçilim söz konusu olsa da tesadüfi başarı ya da başarısızlıklardan öğrenme sürecini arındırabilmek adına bir doğrulama işlemi gerçekleştirmek gereklidir. Bu sayede veri seti dışında gerçek veriler söz konusu olduğunda seçilecek modelin tahmin başarısının yüksek olacağı düşünülebilir. Bu amaçla çapraz doğrulama mekanizması uygulanarak elde edilen sonuçları üzerinde hesaplanan istatistiki değerler yoluyla karar verilmektedir. Şekil 5.12'deki kod içerisinde öncelikle "Kfold" ile "n_split" değeri 40 olarak belirlenerek eğitim verisi 40 farklı parçaya ayrılmıştır. Ardından "cross_val_score" metodu kullanılarak her bir bölümlenmiş veri kullanılarak sınıflandırma modellerinde dikkate alınacak olan "accuracy" skoru üzerinden model değerlendirmeye tabi tutulmak üzere eğitim süreci gerçekleştirilmiştir. Bu noktada sonuç değişkeni üzerinden ".mean()" metodu ile ortalama değer ile modelin başarısı istatistiki açıdan belirlenmeye çalışılmıştır.

"Cross_val_score" metodu ile yukarıda bahsedilen işlemin gerçekleştirilmesi sırasında "kfold" ile bölümlenmiş olan eğitim verisinin her bir bölümünü test verisi olarak, diğerlerini ise eğitim verisi olarak kullanıp tekrar eden bir süreçle eğitim işlemi gerçekleştirilmiştir.

Çapraz doğrulama işlemi sonrası eğitim için ayrılmış ölçeklenmiş verilerle eğitim işlemi gerçekleştirilmiştir. Eğitim işlemi sonrasında başarım oranı 0.4660 çıkmıştır.

```
Lsvc = LinearSVC(random_state=42,multi_class="crammer_singer")
Lsvc.fit(train_set_scaled,y_egit)
y_pred = Lsvc.predict(train_set_scaled)
accuracy_score(y_egit,y_pred)

0.472375
```

Şekil 5.13. Ölçeklenmiş verilerle eğitim işlemi için yazılan kodların ekran görüntüsü

Makine öğrenmesi modellerinin eğitim süreçlerinde matematiksel fonksiyonlar kullanılmaktadır. Bu fonksiyonların kullanımında, değerlerinin geliştirici tarafından atanması gereken ve model başarımını doğrudan etkilen parametrelere hiperparametreler denilmektedir.

Makine öğrenmesi sistemlerinde hiperparametreler kritik bir öneme sahiptir. Farklı hiperparametre değerleri ile yapılan eğitimlerde önemli ölçüde farklı öğrenme performansları elde edilmektedir (Feurer, Springenberg, ve Hutter, 2015). Makine öğrenmesi modellerini farklı problemlere göre uyarlayabilmek edebilmek için hiperparametre optimizasyonu yapılması gerekmektedir. En iyi hiperparametre konfigürasyonun seçilmesinin, modelin performansına doğrudan etkisi bulunmaktadır. Bu seçimleri yapabilmek için hiperparametre optimizasyon teknikleri konusunda derin bir bilgiye sahip olmak gereklidir. Bununla birlikte, optimizasyon amacıyla kullanılan bir çok teknik ve kütüphane desteği de bulunmaktadır (Yang ve Shami, 2020). Bu tez kapsamında kullanılan makine öğrenmesi modellerinde, hiperparametre optimizasyonu için Pedregosa ve arkadaşları tarafından geliştirilen scikit-learn kütüphanesi içerisinde bulunan "RandomizedSearchCV" ve "GridSearchCV" fonksiyonlarından faydalanılmıştır.

"RandomizedSearchCV" fonksiyonu, hiperparametreler üzerinde rastgele aramalar yaparak en iyi hiperparametre değerlerini bulmak için kullanılır. Seçim yapılacak parametreler, bir sözlük ile tanımlanır. Parametrelerin hepsini sırayla taramak yerine rastgele tarama yapması nedeniyle, hesaplama maliyeti düşüktür.

“GridSearchCV” fonksiyonu ile, seçilen tüm hiperparametrelerin olası kombinasyonları sırayla denenmektedir. Her ne kadar bu yöntem ile hesaplama maliyetleri yüksek çıksa da, en iyi hiperparametre kombinasyonunu bulmak için kullanılmaktadır (Pedregosa ve ark., 2011).

Bu çerçeveden bakıldığında, seçilen modelin üstünde eğitim başarımını arttırmak amacıyla öncelikle “RandomizedSearchCV” fonksiyonu ile hiperparametre seçimleri yapılarak sonuç elde edilmesinin ardından, elde edilen sonucun beklenen başarım skorlarının altında olması durumunda, daha yüksek çalışma maliyeti olan ancak tüm kombinasyonların denenmesi ile daha etkili sonuçlar elde edilebilecek “GridSearchCV” fonksiyonunun kullanılmasının uygun olacağı söylenebilir.

Şekil 5.13’de yapılan eğitim sonucu elde edilen başarım skorunun düşük çıkması nedeniyle, “RandomizedSearchCV” fonksiyonu ile parametre seçimi yapılarak elde edilebilecek en yüksek başarım skoru aranmıştır. Bu amaçla yazılan kodların ekran görüntüsü Şekil 5.14’de gösterilmiştir.

```
param_grid = {'C': [0.1, 1, 10, 100, 1000],  }
rndLSVC = RandomizedSearchCV(Lsvc,param_grid)
rndLSVC.fit(train_set_scaled,y_egit)
rndLSVC.best_params_,rndLSVC.best_score_
({'C': 0.1}, 0.465625)
```

Şekil 5.14. RandomizedSearchCV fonksiyonu ile yüksek başarım skoru arama için yazılan kodların ekran görüntüsü

Daha iyi sonuç alabilmek adına “RandomizedSearzhCV” ile yapılan hiperparametre seçimi sonrası 0.465625 değeri elde edilmiştir. Aynı işlem, “GridSearchCV” fonksiyonu ile de denenmiştir.

```
gsLSVC = GridSearchCV(Lsvc, param_grid,cv=5,n_jobs=-1)
gsLSVC.fit(train_set_scaled,y_egit)
gsLSVC.best_params_,gsLSVC.best_score_
({'C': 0.1}, 0.465625)
```

Şekil 5.15. GridSearchCV fonksiyonu ile yüksek başarım skoru arama için yazılan kodların ekran görüntüsü

GridSearchCV fonksiyonu ile en iyi parametrelerin seçilmesi ile de aynı değer elde edilmiştir.

```
LSVC = gsLSVC.best_estimator_
y_preds = LSVC.predict(X)
print(confusion_matrix(y,y_preds))
```

Şekil 5.16. Karmaşıklık matrisi elde edebilmek için yazılan kodların ekran görüntüsü

Optimizasyon sonrasında da skorun düşük olması ile elde edilen karmaşıklık matrisi ekran görüntüsü Şekil 5.17' deki gibidir.

```
[[   0    0    5    0  852   97   18    0]
 [   0    0   12    0  841   95   40    0]
 [   0    0   10    0 1710  204   73    0]
 [   0    0   15    0  823  392  229    0]
 [   0    0    4    0  271  126  132    0]
 [   0    0    0    0    3    7    2    0]
 [   0    0    8    0  146  307  786    0]
 [   0    0   12    0 1153  578 1049    0]]
```

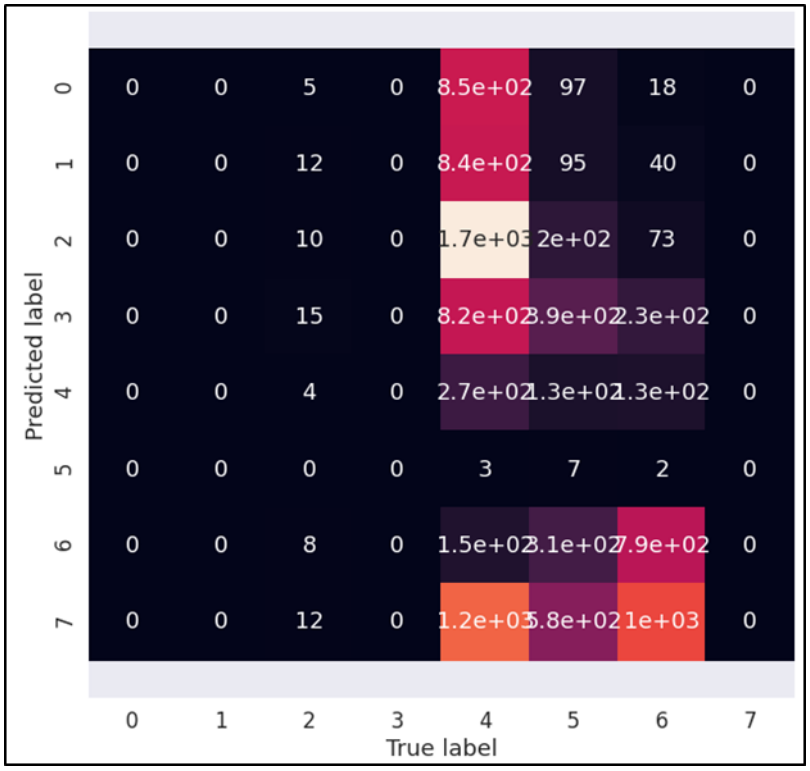
Şekil 5.17. Linear svc ile elde edilen karmaşıklık matrisi

Karmaşıklık matrisi incelendiğinde, örtüşmelerin düşük olması nedeniyle sonuçların tatmin edici olmadığı gözlemlenmiştir.

Elde edilen karmaşıklık matrisinin ısı haritasını oluşturmak için yazılan kodların ekran görüntüsü Şekil 5.18'de, elde edilen ısı haritası ekran görüntüsü ise Şekil 5.19'de görülmektedir.

```
sns.set(font_scale=1.5)
def plot_conf_mat(y_test, y_preds):
    fig, ax = plt.subplots(figsize=(10, 10))
    ax = sns.heatmap(confusion_matrix(y_test, y_preds),
                     annot=True,
                     cbar=False)
    plt.xlabel("True label")
    plt.ylabel("Predicted label")
    bottom, top = ax.get_ylim()
    ax.set_ylim(bottom + 0.5, top - 0.5)
plot_conf_mat(y, y_preds)
```

Şekil 5.18. Karmaşıklık matrisi ısı haritası oluşturmak için yazılan kodların ekran görüntüsü



Şekil 5.19. Doğrusal destek vektör sınıflandırıcısı ile eğitim işlemi sonrası karmaşıklık matrisi ısı haritası

Doğrusal destek vektör sınıflandırıcısı ile yapılan eğitim ve optimizasyon işlemleri sonrasında elde edilen en yüksek başarım skorunun düşük olması (0.4656) olması nedeniyle, Şekil 5.19 'da görülen ısı haritasında tahmin doğruluğunun da düşük olduğu gözlemlenmiştir. Isı haritası incelendiğinde tahmin edilen bazı sonuçların veri setindeki olması gereken gerçek değerle örtüşmediği tespit edilmiştir.

Başarım skorunun düşük olması ve çok sınıflı problemlerde diğer metriklerin de göz önüne alınması gerektiğinden, Şekil….'daki kodlar yazılarak diğer metriklerdeki sonuçlar da incelenmiştir.

```
from sklearn.metrics import precision_score,recall_score,f1_score,accuracy_score
accuracy_score(y,y_preds)
0.1355

recall_score(y,y_preds,average="weighted")
0.1355

precision_score(y,y_preds,average="micro")
0.1355

f1_score(y,y_preds,average="micro")
0.1355
```

Şekil 5.20. Linear svc diğer metriklerin sonuçları ekran görüntüsü

Şekil 5.20'de görülmekte olan diğer metrikler de incelendiğinde skorların düşük olduğu görülmüştür. Linear SVC makine öğrenme modeli çok sınıflamalı problemlerin hipotezlerinin hesaplanmasında doğruluk, eğitim zamanı açısında etkili ve başarı olsa da örneklem (veri seti kayıt sayısı) sayısının özellik sayısından (veri seti sütun sayısı) çok fazla olması durumunda başarısız olabilme ihtimali yüksektir. Bu noktadan bakıldığında destek vektör sınıflandırıcısı kullanılarak yapılan eğitim işleminden sonra elde edilen bulgular, bu makine öğrenmesi modelinin üretilen veri setindeki veriler temel alınarak, yıkım teknikleri seçimi sınıflandırılmasında kullanılmasının uygun olmadığı değerlendirilmiştir.

Pedregosa ve arkadaşları tarafından geliştirilen makine öğrenmesi model seçimi yol haritası takip edildiğinde, doğrusal destek vektör sınıflandırıcısı algoritmasından elde edilen sonucun düşük çıkması nedeniyle, veri setindeki verilerin niteliğine uygun sonraki model olan k en yakın komşu algoritması ile de eğitim işlemi gerçekleştirilmiştir.

5.2.2. K en yakın komşu algoritması (Kneighbors algoritması) için geliştirilen kodlar ve sonuçları

K en yakın komşu algoritması ile makine öğrenimini gerçekleştirmek üzere veri görselleştirme ve analiz sürecinde kullanılan sınıflar, modelin ve veri setinin hazırlanmasından modele ait hiper parametrelerin seçim yapılması ile ilgili kütüphanelerin

tamamı mevcut kod sayfasına eklenmiştir. Eklenen kodların ekran görüntüsü Şekil 5.21' de gösterilmiştir.

```
      import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import MinMaxScaler as Scaler
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.model_selection import RandomizedSearchCV,GridSearchCV
from sklearn.metrics import confusion_matrix,classification_report
from sklearn.metrics import precision_score,recall_score,f1_score
from sklearn.metrics import roc_curve,plot_confusion_matrix,plot_roc_cur
ve
bilgi = pd.read_csv("data.csv")
```

Şekil 5.21. K en yakın komşu algoritması süreçleri için kullanılan kütüphaneler ekran görüntüsü

Veri setinin elde edilmesinin ardından veriler üstünde eğitim ve test verileri olmak üzere bölme işlemi ve ölçeklendirme işlemi yapılmıştır. Aynı zamanda ölçeklendirme yapılarak makine öğrenmesinin gerçekleştirilmesi sürecinde veriler arasındaki maksimum ve minimum değerler arasındaki değişim aynı ölçekle dikkate alınması sağlanmıştır.

Yapılan işleme ait kodlar'ın ekran görüntüsü Şekil 5.22' de gösterilmiştir.

```
X = bilgi.drop("sonuc",axis=1)
y = bilgi["sonuc"]
X_egit,X_test,y_egit,y_test = train_test_split(X,y,test_size=0.2)
scaler = Scaler()
scaler.fit(X_egit)
train_set_scaled = scaler.transform(X_egit)
test_set_scaled = scaler.transform(X_test)
scaled_X = scaler.transform(X)
```

Şekil 5.22. Verilerin eğitim ve test verisi olarak bölünmesi ve ölçeklendirme işlemi ekran görüntüsü

Verilerin eğitim ve test verileri olarak bölünmesinden sonra, k en yakın komşu algoritması kullanılarak eğitim işlemi gerçekleştirilmiştir. Bununla birlikte eğitim için K en yakın komşu algoritmasının varsayılan eğitim parametreleri kullanılmıştır. Eğitim işlemi ve çıktısının ekran görüntüsü Şekil 5.23' de belirtilmiştir.

```
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X,y)

KNeighborsClassifier(algorithm='auto', leaf_size=30,
metric='minkowski',metric_params=None, n_jobs=None, n_neighbors=1,
p=2,
weights='uniform')

knn.score(test_set_scaled,y_test)

0.4515
```

Şekil 5.23. K en yakın komşu algoritması ile eğitim işlemi ve çıktısı

Şekil 5.23 'de görülen eğitim çıktısına göre, veriler arasındaki uzaklıkları hesaplamak için minkowski yönteminin kullanıldığı görülmektedir. K en yakın komşu algoritmasında, komşu verileri arasındaki uzaklık hesaplama yöntemleri, eculidean, manhattan ve minkowski'dir (Bhatnagar ve Srivastava, 2019). Bu parametre değerleri ile 0.4515 skoru elde edilmiştir ve düşük bir skor olarak değerlendirilmiştir.

Eğitim işleminin tamamlanmasının ardından K en yakın komşu algoritmasından elde edilen skoru arttırmak amacıyla RandomizedSearchCV fonksiyonu ile, K en yakın komşu algoritmasının rastgele verilen hiper paremetrelerin hangisiyle en iyi sonucu vereceği araştırılmıştır. Bu işlem için hiper parametreler bir grid oluşturacak Şekilde sıralanmıştır. RandomizeSearchCV (Rastgele arama çapraz doğrulama) fonksiyonu ile en iyi sonuç verecek hiperparametre seti elde edilmeye çalışılmıştır. Bu işlem için yirmi beş iterasyonla, ölçeklendirilmiş eğitim verisi çapraz doğrulama için beş'e bölünmüştür. Yazılan kodlar Şekil 5.24' deki ekran görüntüsünde gösterilmiştir.

```
n_neighbors = [int(x) for x in np.linspace(start = 1, stop = len(
X_egit)/3, num = 5)]
weights = ['uniform', 'distance']
algorithm  = ["auto","ball_tree","kd_tree","brute"]
leaf_size  = [int(x) for x in np.linspace(10, 100, num = 5)]
p  = [1,2]

random_grid = {'n_neighbors': n_neighbors,
               'weights': weights,
               'algorithm': algorithm,
               'leaf_size': leaf_size,
               'p': p}

knn_clf = KNeighborsClassifier()
knn_random = RandomizedSearchCV(estimator = knn_clf, param_distribut
ions = random_grid, n_iter = 25, cv = 5, verbose=1,)
knn_random.fit(train_set_scaled, y_egit)

Fitting 5 folds for each of 25 candidates, totalling 125 fits
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent
workers.
[Parallel(n_jobs=1)]: Done 125 out of 125 | elapsed:  2.2min finished
RandomizedSearchCV(cv=5, error_score=nan,
                   estimator=KNeighborsClassifier(algorithm='auto',
                                                  leaf_size=30,
                                                  metric='minkowski',
                                                  metric_params=None,
                                                  n_jobs=None,
n_neighbors=5,
                                                  p=2,
weights='uniform'),
                   iid='deprecated', n_iter=25, n_jobs=None,
                   param_distributions={'algorithm': ['auto',
'ball_tree',
                                                      'kd_tree',
'brute'],
                                        'leaf_size': [10, 32, 55, 77,
100],
                                        'n_neighbors': [1, 667, 1333,
2000,
                                                        2666],
                                        'p': [1, 2],
                                        'weights': ['uniform',
'distance']},
                   pre_dispatch='2*n_jobs', random_state=None,
refit=True,
                   return_train_score=False, scoring=None, verbose=1)

knn_random.best_score_,knn_random.best_estimator_,knn_random.best_params
_
```

Şekil 5.24. K en yakın komşu algoritmasında RandomizedSearchCV fonksiyonu ile en iyi hiper parameterleri bulmak amacıyla yazılımış kodların ekran görüntüsü

Yürütülen işlem sonucunda elde edilen en iyi skor, en iyi model ve en iyi hiper parametereler listelenerek ekran görüntüsü Şekil 5.24’ de gösterilmiştir. İşlem süreci 2.2 dakika sürmüştür ve her biri 25 aday veriden oluşan 5 katmanla toplam 125 veri üstünde eğitim gerçekleşmiştir.

```
(0.47725,
 KNeighborsClassifier(algorithm='auto', leaf_size=32,
metric='minkowski',metric_params=None, n_jobs=None,
n_neighbors=1333, p=1,weights='distance'),
 {'algorithm': 'auto',
  'leaf_size': 32,
  'n_neighbors': 1333,
  'p': 1,
  'weights': 'distance'})
```

Şekil 5.25. RandomizeSearchCV fonksiyonu ile elde edilen sonuçların ekran görüntüsü

RandomizeSearchCV fonksiyonundan elde edilen minkowski metriği ve en iyi parametrelere göre elde edilen en iyi başarım skoru 0.47725 olmuştur.

RandomizeSearchCV fonksiyonu ile elde edilen başarım skorunun düşük olması sonucundan yola çıkılarak, bütün parametreleri ölçeklendirilmiş eğitim verisi 5’e bölünerek ve bir önceki işlemde kullanılan aynı hiper parametre listesi kullanılarak GridSearchCV fonksiyonu ile eğitim işlemi gerçekleştirilmiştir. Eğitim işlemi için yazılan kodlar Şekil 5.26’ da gösterilmiştir.

```
n_neighbors = [int(x) for x in np.linspace(start = 1, stop = len(X_e
git)/3, num = 5)]
weights = ['uniform', 'distance']
algorithm  = ["auto","ball_tree","kd_tree","brute"]
leaf_size  = [int(x) for x in np.linspace(10, 100, num = 5)]
p  = [1,2]

grid_search = {'n_neighbors': n_neighbors,
               'weights': weights,
               'algorithm': algorithm,
               'leaf_size': leaf_size,
               'p': p}

knn_clf = KNeighborsClassifier()
knn_grid = GridSearchCV(estimator = knn_clf, param_grid=grid_search,
 cv = 5, verbose=1,)
knn_grid.fit(train_set_scaled, y_egit)

Fitting 5 folds for each of 400 candidates, totalling 2000 fits
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent
workers.
[Parallel(n_jobs=1)]: Done 2000 out of 2000 | elapsed: 30.4min finished
GridSearchCV(cv=5, error_score=nan,
             estimator=KNeighborsClassifier(algorithm='auto',
leaf_size=30,
                                            metric='minkowski',
                                            metric_params=None,
n_jobs=None,
                                            n_neighbors=5, p=2,
                                            weights='uniform'),
             iid='deprecated', n_jobs=None,
             param_grid={'algorithm': ['auto', 'ball_tree', 'kd_tree',
'brute'],
                         'leaf_size': [10, 32, 55, 77, 100],
                         'n_neighbors': [1, 667, 1333, 2000, 2666], 'p':
[1, 2],
                         'weights': ['uniform', 'distance']},
             pre_dispatch='2*n_jobs', refit=True,
return_train_score=False,
             scoring=None, verbose=1)

knn_grid.best_score_,knn_grid.best_estimator_,knn_grid.best_params_
```

Şekil 5.26. K en yakın komşu algoritmasında GridSearchCV fonksiyonu ile en iyi hiper parameterleri bulmak amacıyla yazılımış kodların ekran görüntüsü

Yürütülen işlem sonucunda elde edilen en iyi skor, en iyi model ve en iyi hiper parametereler listelenmiştir. İşlem süreci 30.4 dakika sürmüştür ve her biri 400 aday veriden oluşan 5 katmanla toplam 2000 veri üstünde eğitim gerçekleşmiştir.

```
(0.48687500000000006,
 KNeighborsClassifier(algorithm='kd_tree', leaf_size=77,
metric='minkowski',
                      metric_params=None, n_jobs=None,
n_neighbors=667, p=1,
                      weights='distance'),
 {'algorithm': 'kd_tree',
  'leaf_size': 77,
  'n_neighbors': 667,
  'p': 1,
  'weights': 'distance'})
```

Şekil 5.27. GridSearchCV fonksiyonu ile elde edilen sonuçların ekran görüntüsü

GridSearchCV fonksiyonu ile minkowski metriğine ve en iyi parametrelere göre elde edilen en yüksek başarım skoru 0.486875 olmuştur. Yürütülen işlemler doğrultusunda elde edilen sonuç sadece eğitim verisi üzerinden gerçekleştirilmiştir. Bu nedenle başarım skoru genel olarak verilmiş bir başarım skorudur. Sınıflandırma modellerinde dikkate alınacak olan metrikler açısından değerlendirilmesi bu süreçte eğitim ve test verisinin birleştirilerek ölçeklendirilmesi sonrasında elde edilen modele yönelik değerlendirmenin daha gerçekçi bir sonuç vereceği değerlendirilebilir.

```
model = knn_grid.best_estimator_
y_preds = knn_grid.predict(scaled_X)
print(confusion_matrix(y_preds,y))
```

Şekil 5.28. En iyi skora sahip k en yakın komşu modeline göre karmaşıklık matrisi üretmek için yazılan kodlar

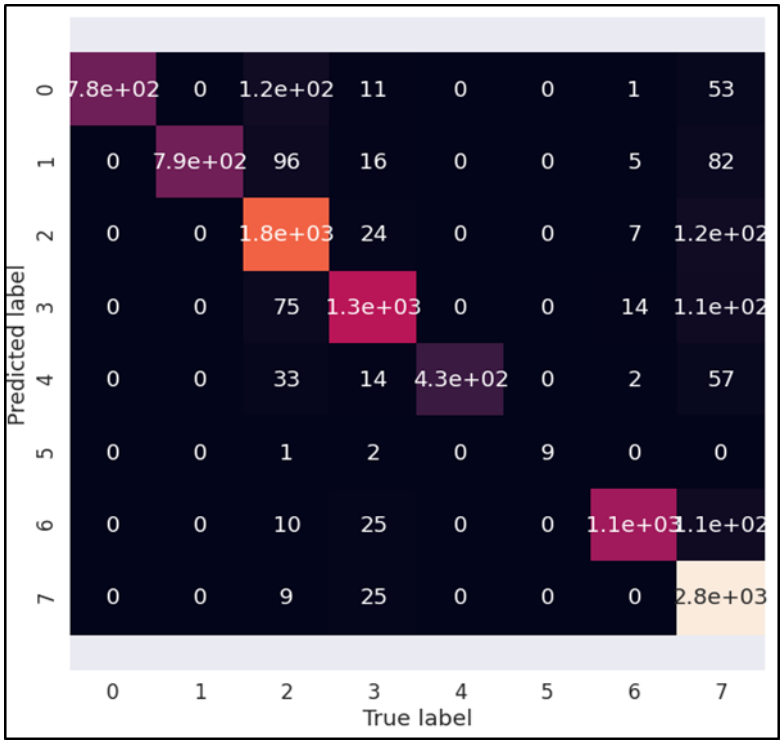
Yürütülen işlem sonucunda elde edilen en iyi skora sahip K en yakın komşu modeline bütün eğitim verileri göndererek elde edilen tahminler (y_preds), veri setindeki gerçek sonuç verileriyle karşılaştırılmıştır (y). Bu karşılaştırma sonrasında elde edilen karmaşıklık matrisi ekran görüntüsü Şekil 5.29'de görülmektedir.

```
[[ 785    0    0    0    0    0    0    0]
 [   0  789    0    0    0    0    0    0]
 [ 122   96 1844   75   33    1   10    9]
 [  11   16   24 1257   14    2   25   25]
 [   0    0    0    0  427    0    0    0]
 [   0    0    0    0    0    9    0    0]
 [   1    5    7   14    2    0 1102    0]
 [  53   82  122  113   57    0  110 2758]]
```

Şekil 5.29. En iyi skora sahip K en yakın komşu modeline gönderilen eğitim verileri sonrası tahminler ve veri setindeki gerçek değerlerin örtüşme oranları

```
sns.set(font_scale=1.5)

def plot_conf_mat(y_test, y_preds):
    fig, ax = plt.subplots(figsize=(10, 10))
    ax = sns.heatmap(confusion_matrix(y_test, y_preds),
                     annot=True,
                     cbar=False)
    plt.xlabel("True label")
    plt.ylabel("Predicted label")

    bottom, top = ax.get_ylim()
    ax.set_ylim(bottom + 0.5, top - 0.5)

plot_conf_mat(y, y_preds)
```

Şekil 5.30. K en yakın komşu modeli tahminleri ve gerçek değerleri karşılaştırması için ısı haritası oluşturmak adına yazılan kodların ekran görüntüsü



Şekil 5.31. K en yakın komşu modeli karmaşıklık matrisi ısı haritası

Bu çalışmada problem yapısı incelendiğinde kullanılan makine öğrenme modelleri gözetimli öğrenme ve sınıflandırmaya yönelik modellerdir. Yukarıdaki sonuçlar ele alındığında KNN modelinin başarısız olduğu düşünülebilir. Modelin başarımını ölçmek üzere kullanılan farklı metrikler de incelenerek modelin faydalı olup olmadığı kararının gerekçeleri ve dayanakları arttırılabilir. Bu aşamada eğitilmiş modelden elde edeceğimiz değerin bağımsız değişkenlere uyumu 4 durumda incelenebilir;

- True Positive : Modelin doğru tahmin ettiği olumlu durumdur.
- True Negative : Modelin doğru tahmin ettiği olumsuz durumdur.
- False Positive : Modelin yanlış tahmin ettiği olumlu durumdur.
- False Negative : Modelin yanlış tahmin ettiği olumsuz durumdur.

Dört Durum üzerinden aşağıda değerleri alınan metrikler incelendiğinde ;

- Accuracy Metriği : Doğru tahmin sayısı /Tüm tahmin sayısı
- Precision Metriği : Olumlu tahminlerin kaçının gerçek olduğu değeridir.
- True Positive / (True Positive + False Positive)
- Recall Metriği : Olumlu durumları yakalanma oranı
- True Positive / (True Positive + False Positive)
- F1 Score: Precision Metriği ve Recall Metriğinin Harmonik ortalamasıdır.

Bu noktada verisetinin yapısına ve problem durumuna göre farklı metriklerin dikkate alınması gerektiği durumlar göz önünde bulundurulmalıdır. Çalışmada yıkım karar destek sistemi ve birden fazla sınıflamalı problem ele alındığında F1 Score değerinin Accuracy Score değerine göre daha kullanışlı olduğu söylenilebilir.

```
from sklearn.metrics import precision_score,recall_score,f1_score,accuracy_score
accuracy_score(y,y_preds)
0.8971
recall_score(y,y_preds,average="weighted")
0.8971
precision_score(y,y_preds,average="micro")
0.8971
f1_score(y,y_preds,average="micro")
0.8971
```

Şekil 5.32. K en yakın komşu modelinden elde edilen başarım skorları

Tüm eğitim verilerinin ölçeklendirilerek K en yakın komşu algoritması ile eğitilmesi sonucunda sınıflandırıcı modellerde dikkate alınabilecek metrikler üzerinden elde edilen başarım skoru ise 0.8971 olarak tespit edilmiştir. Elde edilen yüksek başarım skoru neticesinde oluşan karmaşıklık matrisi ısı haritası incelendiğinde, tahmin edilen değerlerin, veri setinde etiketlenmiş gerçek değerlerle büyük oranda örtüştüğü gözlemlenmiştir. Bu bulgu ışığında, k en yakın komşu algoritması ile eğitilen sentetik veri setinin, yıkım teknikleri seçiminde yol gösterici olarak kullanılabileceği düşünülmektedir. Aynı zamanda mevcut ısı haritası üretilen sentetik veri üzerinden etiketlerin yoğunluğu dikkate alındığında en az örtüşmenin görüldüğü 5 durumu için 174 veri üzerinden 9 doğru tahmin elde edildiği görülmüştür. Elde edilen başarım skoru her ne kadar yüksek de olsa, Pedregosa ve arkadaşlarının önerdiği makine öğrenmesi model seçimi yol haritası takip edilerek, destek vektör sınıflandırıcısı makine öğrenmesi algoritması ile de eğitim işlemi gerçekleştirilmiştir.

5.2.3. Destek vektör sınıflandırcısı (Support vector classifier – Svc) için geliştirilen kodlar ve sonuçları

Destek vektör sınıflandırıcısı, support vector machine (svm) makine öğrenmesi algoritmasının, python programlama dili için Pedregosa ve arkadaşları tarafından geliştirilen scikit learn kütüphanesinde sınıflandırma problemlerine yönelik kullanılan modelin ismidir. Bu modelin kullanımı için Şekil 5.33’ de görülen kütüphanelerden faydalanılmıştır.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.svm import SVC
from sklearn.preprocessing import MinMaxScaler as Scaler
from sklearn.model_selection import train_test_split,cross_val_score,cro
ss_val_predict
from sklearn.model_selection import RandomizedSearchCV,GridSearchCV
from sklearn.metrics import confusion_matrix,classification_report
import sklearn.metrics as metrics
from sklearn.metrics import precision_score,recall_score,f1_score,ac
curacy_score
from sklearn.metrics import roc_curve,plot_confusion_matrix,plot_roc_cur
ve
bilgi = pd.read_csv("data.csv")
```

Şekil 5.33. Svc makine öğrenmesi modelini kullanabilmek için gerekli olan kütüphaneler içi yazılan kodların ekran görüntüsü

Modelin değerlendirilmesinde diğer modellerde kullanılan veri seti kullanılmıştır.

	yukseklik	tasiyici	kullanim	durum	hasar	komsu	kamubina	sudogalgaz	gurultu	toz	titresim	yolmesafe	sonuc
0	45	5	2	1	1	60	0	1	36	4	568	80	2
1	48	4	6	1	2	15	1	0	27	1	503	55	7
2	12	1	2	1	3	5	0	1	35	4	528	60	1
3	51	5	1	0	1	100	1	1	75	2	394	25	6
4	42	3	4	0	2	80	1	1	45	3	561	35	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...
9995	39	2	3	0	2	35	0	0	43	4	535	20	3
9996	39	1	5	1	2	45	0	1	21	1	215	20	7
9997	48	2	3	1	2	20	1	1	34	3	413	15	7
9998	27	1	1	1	2	85	0	0	20	3	484	70	2
9999	36	1	4	1	1	100	0	0	27	2	518	95	3

10000 rows × 13 columns

Şekil 5.34. Üretilen sentetik veri seti ekran görüntüsü

Veri setinin eğitime hazırlanabilmesi için, veri seti üstünde ölçeklendirme yapılarak eğitim ve test verileri olarak ayrıştırma işlemi yapılmıştır. Bu işlem için yazılan kodlar Şekil 5.35'de gösterilmiştir.

```
X = bilgi.drop("sonuc",axis=1)
y = bilgi["sonuc"]
X_egit,X_test,y_egit,y_test = train_test_split(X,y,test_size=0.2)
scaler = Scaler()
scaler.fit(X_egit)
train_set_scaled = scaler.transform(X_egit)
test_set_scaled = scaler.transform(X_test)
scaled_X = scaler.transform(X)
```

Şekil 5.35. Veri üstünde yapılan ölçeklendirme işlemi ve verilerin eğitim-test verisi olarak bölünmesi için yazılan kodların ekran görüntüsü

Verilerin ölçeklendirilmesi işlemi sonrasında k katmanlı çapraz doğrulama işlemi, ölçeklenmiş veriler üzerinde çalıştırılarak, tesadüfi başarıların önüne geçilmesinin yanı sıra, birden fazla eğitim-test grubunda modelin performansına yönelik daha fazla bilgi elde edilmiştir. Aynı zamanda iyileştirme çalışmalarında kullanabilmek adına hiper parametre listesi de elde edilmiştir. Bu işlem için yazılan kodlar ve sonuçları Şekil 5.36' da gösterilmiştir.

```
svc = SVC()
print(*svc.get_params(),sep="\n")
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
kfold = KFold(n_splits=10)
sonuc = cross_val_score(svc, scaled_X, y, cv=kfold, scoring='accurac
y')
print(sonuc.mean())

C
break_ties
cache_size
class_weight
coef0
decision_function_shape
degree
gamma
kernel
max_iter
probability
random_state
shrinking
tol
verbose
0.6802
```

Şekil 5.36. K katmanlı çapraz doğrulama işlemi yazılan kodlar ve elde edilen sonuçların ekran görüntüsü

Veri seti birden fazla etikete sahip olduğundan çok sınıflı bir sınıflandırma problemini çözüm adına fonksiyon oluşturulmuştur. Bu fonksiyon ile sonuçlarla ilgili olarak metrikler üzerinden bilgiler elde edilecektir. Bu bilgiler arasında her bir sınıf üzerindeki başarım adına sonuçlar da elde edilmiştir. Yazılan fonksiyon ekran görüntüsü Şekil 5.37'de gösterilmiştir.

```
def multi_class_classification(data_X,data_Y):
    svc = SVC(C=1, kernel='linear')
    clf = svc.fit(data_X, data_Y) #svm
    predicted = cross_val_predict(clf, data_X, data_Y, cv=2)
    print("accuracy",metrics.accuracy_score(data_Y, predicted))
    print("f1 score macro",metrics.f1_score(data_Y, predicted, avera
ge='macro'))
    print("f1 score micro",metrics.f1_score(data_Y, predicted, avera
ge='micro'))
    print("precision score",metrics.precision_score(data_Y, predicte
d, average='macro'))
    print("recall score",metrics.recall_score(data_Y, predicted, ave
rage='macro'))
    print("hamming_loss",metrics.hamming_loss(data_Y, predicted))
    print("classification_report", metrics.classification_report(dat
a_Y, predicted))
    print("jaccard_similarity_score", metrics.jaccard_similarity_sco
re(data_Y, predicted))
    print("zero_one_loss", metrics.zero_one_loss(data_Y, predicted))
    return clf
```

Şekil 5.37. Çok sınıflı problemin çözümüne yönelik yazılan fonksiyonun ekran görüntüsü

Yazılan fonksiyon kullanıldığında yapılan eğitim işlemi ile elde edilen başarım skoru 0.6039 olmuştur. Elde edilen skor ve sonuçlar Şekil 5.37' da gösterilmiştir. Başarım skoru değerlendirildiğinde, düşük bir skor olduğunu düşünülmektedir.

```
clf = multi_class_classification(X,y)

accuracy 0.6039
f1 score macro 0.4040665746841251
f1 score micro 0.6039
precision score 0.5116696926966695
recall score 0.4389586856567782
hamming_loss 0.3961
classification_report               precision    recall  f1-score
support

           0       0.60      0.78      0.68       972
           1       0.59      0.02      0.04       988
           2       0.65      0.75      0.70      1997
           3       0.42      0.46      0.44      1459
           4       0.57      0.01      0.01       533
           5       0.00      0.00      0.00        12
           6       0.59      0.70      0.64      1247
           7       0.67      0.79      0.73      2792

    accuracy                           0.60     10000
   macro avg       0.51      0.44      0.40     10000
weighted avg       0.60      0.60      0.56     10000

jaccard_similarity_score 0.6039
zero_one_loss 0.3961
```

Şekil 5.38. Svc ile eğitim işleminden sonra elde edilen sonuçlar

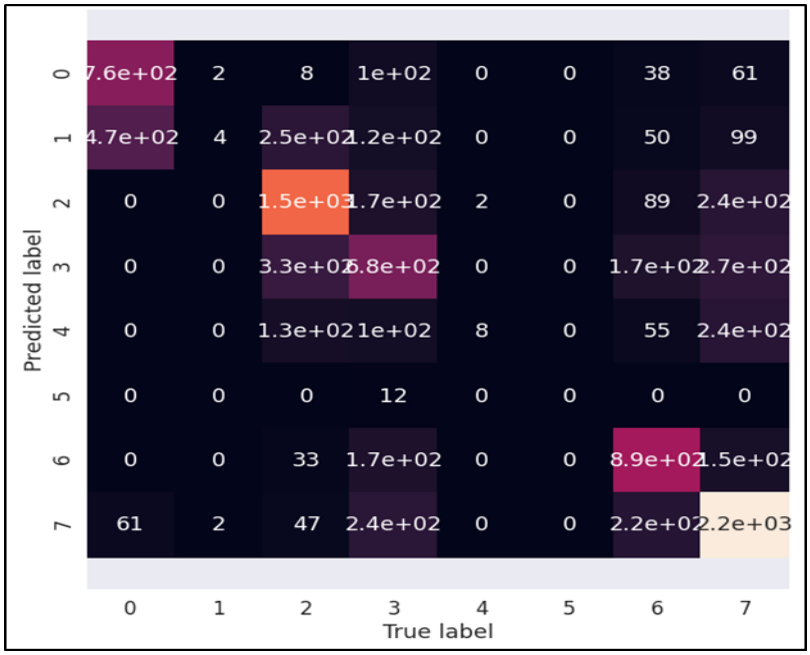
Eğitim işlemi sonrasında, karmaşıklık matrisi oluşturularak gerçek verilerle tahmin verilerinin örtüşme seviyeleri değerlendirilmiştir. Karmaşıklık matrisi ve ısı haritası için yazılan kodların ekran görüntüsü Şekil 5.39'de, elde edilen karmaşıklık matrisi ekran görüntüsü Şekil 5.40'de, ısı haritası ise Şekil 5.41' da gösterilmiştir.

```
y_preds = clf.predict(scaled_X)
print(confusion_matrix(y_preds,y))
sns.set(font_scale=1.5)
def plot_conf_mat(y_test, y_preds):
    fig, ax = plt.subplots(figsize=(10, 10))
    ax = sns.heatmap(confusion_matrix(y_test, y_preds),
                     annot=True,
                     cbar=False)
    plt.xlabel("True label")
    plt.ylabel("Predicted label")

    bottom, top = ax.get_ylim()
    ax.set_ylim(bottom + 0.5, top - 0.5)
plot_conf_mat(y, y_preds)
```

Şekil 5.39. Svc ile yapılan eğitim sonrası karmaşıklık matrisi ve ısı haritası elde etmek için yazılan kodların ekran görüntüsü

```
[[ 760  468    0    0    0    0    0   61]
 [   2    4    0    0    0    0    0    2]
 [   8  252 1499  331  126    0   33   47]
 [ 103  115  166  682  104   12  173  243]
 [   0    0    2    0    8    0    0    0]
 [   0    0    0    0    0    0    0    0]
 [  38   50   89  174   55    0  894  224]
 [  61   99  241  272  240    0  147 2215]]
```

Şekil 5.40. Svc ile yapılan eğitim sonucu elde edilen karmaşıklık matrisi ekran görüntüsü



Şekil 5.41. Svc ile yapılan eğitim sonucu elde edilen karmaşıklık matrisi ısı haritası ekran görüntüsü

Şekil 5.41'daki ısı haritası incelendiğinde, tahmin edilen değerler ve gerçek değerler arasındaki örtüşme seviyelerinin düşük olduğu gözlemlenmiştir. Bu anlamda skoru daha iyi hale getirebilmek adına GridSearchCV fonksiyonu ile en iyi parametrelerle işlem tekrar denenmiştir. Bu işlem için yazılan kodların ekran görüntüsü Şekil 5.42.'da gösterilmiştir.

```
def multi_class_classificationwGridSearchCV(data_X,data_Y):
      param_grid = {'C':[1,10,100,1000],'gamma':[1,0.1,0.001,0.0001]
, 'kernel':['linear','rbf']}
    svc = GridSearchCV(SVC(),param_grid,refit = True, verbose=2)
    clf = svc.fit(data_X, data_Y) #svm
    print(clf.best_params_)
    print(clf.best_score_)
    predicted = cross_val_predict(clf, data_X, data_Y, cv=2)
    print("accuracy",metrics.accuracy_score(data_Y, predicted))
    print("f1 score macro",metrics.f1_score(data_Y, predicted, avera
ge='macro'))
    print("f1 score micro",metrics.f1_score(data_Y, predicted, avera
ge='micro'))
    print("precision score",metrics.precision_score(data_Y, predicte
d, average='macro'))
    print("recall score",metrics.recall_score(data_Y, predicted, ave
rage='macro'))
    print("hamming_loss",metrics.hamming_loss(data_Y, predicted))
    print("classification_report\n", metrics.classification_report(d
ata_Y, predicted))
    print("jaccard_similarity_score", metrics.jaccard_similarity_sco
re(data_Y, predicted))
    print("zero_one_loss", metrics.zero_one_loss(data_Y, predicted))
    return clf

clf = multi_class_classificationwGridSearchCV(scaled_X,y)
[Parallel(n_jobs=1)]: Done 160 out of 160 | elapsed: 13.2min
finished {'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'} 0.873
```

Şekil 5.42. GridSearchCV fonksiyonu ile en iyi parametrelerle yapılan optimizasyon ve eğitim işlemi için yazılan kodların ekran görüntüsü

13.2 dakika süren işlem sonucunda elde edilen skor 0.873 olarak ölçülmüştür. Buna bağlı olarak elde edilen karmaşıklık matrisi için yazılan kodların ekran görüntüsü Şekil 5.43'de, elde edilen karmaşıklık matrisi ise Şekil 5.44'de gösterilmiştir.

```
y_preds = clf.predict(scaled_X)
print(confusion_matrix(y_preds,y))
```

Şekil 5.43. Optimizasyon sonrası yapılan eğitim sonucu karmaşıklık matrisi için yazılan kodların ekran görüntüsü

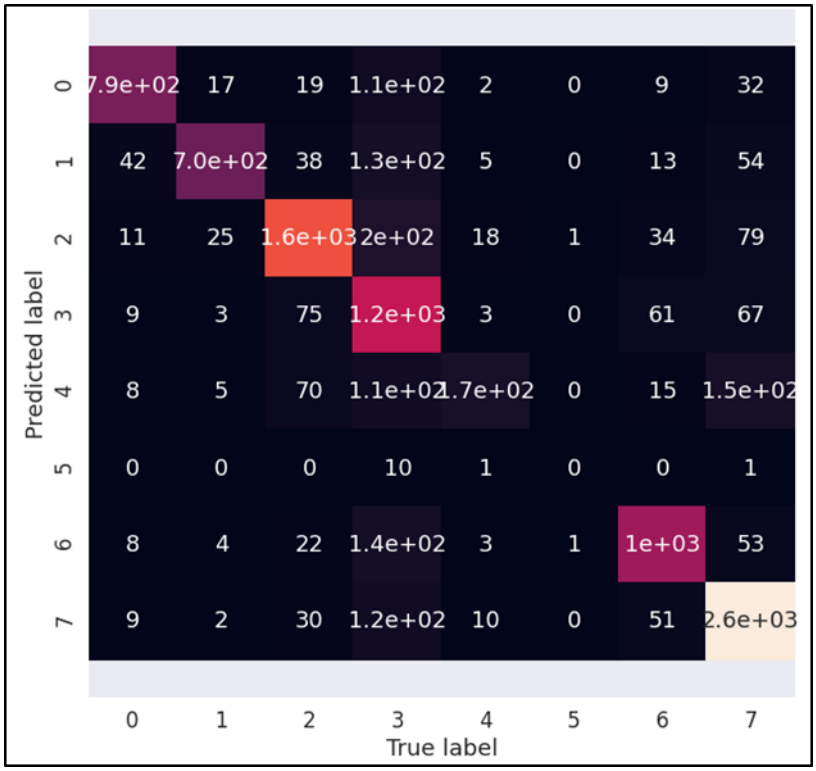
```
[[ 787   42   11    9    8    0    8    9]
 [  17  705   25    3    5    0    4    2]
 [  19   38 1631   75   70    0   22   30]
 [ 106  131  198 1241  114   10  136  116]
 [   2    5   18    3  167    1    3   10]
 [   0    0    1    0    0    0    1    0]
 [   9   13   34   61   15    0 1020   51]
 [  32   54   79   67  154    1   53 2574]]
```

Şekil 5.44. Optimizasyon sonrası yapılan eğitim sonucu karmaşıklık matrisi ekran görüntüsü

Karmaşıklık matrisinin okunmasını kolaylaştırmak adına ısı haritası oluşturma için yazılan kodların ekran görüntüsü Şekil 5.45'de, kodların çalıştırılması sonrası elde edilen ısı haritası ekran görüntüsü ise Şekil 5.46'de gösterilmiştir.

```
y_preds = clf.predict(scaled_X)
print(confusion_matrix(y_preds,y))

sns.set(font_scale=1.5)

def plot_conf_mat(y_test, y_preds):
    fig, ax = plt.subplots(figsize=(10, 10))
    ax = sns.heatmap(confusion_matrix(y_test, y_preds),
                     annot=True,
                     cbar=False)
    plt.xlabel("True label")
    plt.ylabel("Predicted label")

    bottom, top = ax.get_ylim()
    ax.set_ylim(bottom + 0.5, top - 0.5)

plot_conf_mat(y, y_preds)
clf.best_params_
```

Şekil 5.45. Karmaşıklık matrisi ısı haritası elde etmek için yazılan kodların ekran görüntüsü



Şekil 5.46. Karmaşıklık matrisi ısı haritası

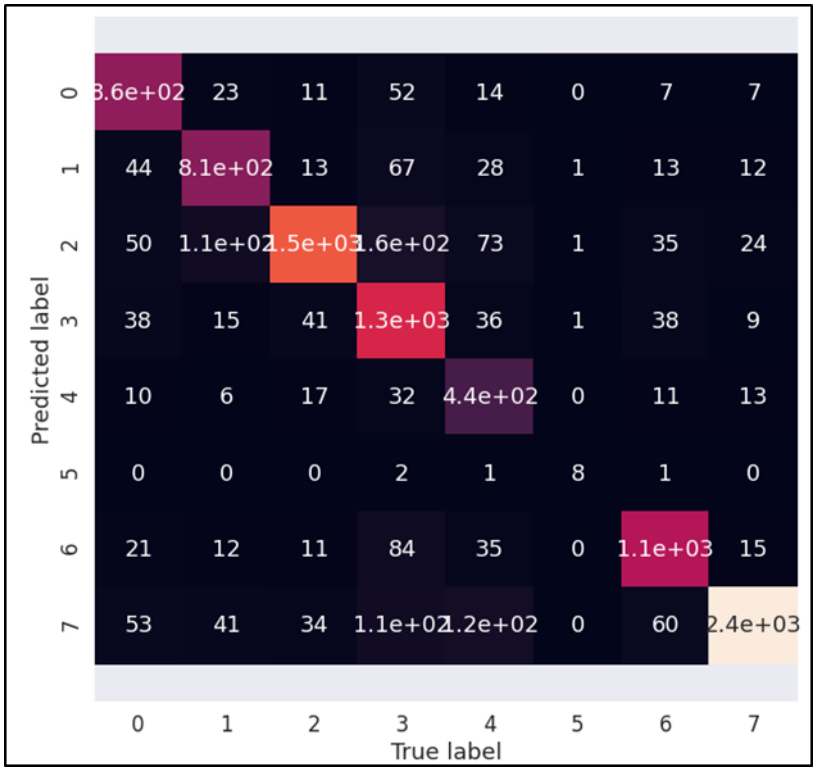
Şekil 5.46'de gösterilen ısı haritası incelendiğinde tahmin değerleri ve gerçek değerler arasındaki örtüşme seviyelerinin düşük olduğu noktalar gözlemlenmiştir. Bu noktadan hareketle veriler ölçeklenmiş veriler ile tekrar eğitim işlemi gerçekleştirilmiştir. Ölçeklendirme işlemi ile verilerin aynı aralıklara getirilerek daha etkili eğitim işlemi yapılması hedeflenmiştir. Eğitim işleminin ardından tekrar karmaşıklık matrisi ve ısı haritası elde edilerek eğitim işlemi sonrası tahmin değerlerinin ve gerçek değerlerin örtüşme oranları yeniden değerlendirilmiştir. Ölçeklendirilmiş veri ile yapılan eğitim ve karmaşıklık matrisi oluşturmak için yazılan kodların ekran görüntüsü Şekil 5.47' de, karmaşıklık matrisi ekran görüntüsü Şekil 5.48' da, ısı haritası ekran görüntüsü ise Şekil 5.49' de gösterilmiştir.

```
svc = SVC(C=1000, gamma=0.1, kernel='rbf',probability=True,class_wei
ght='balanced')
svc.fit(train_set_scaled,y_egit)
y_preds = svc.predict(scaled_X)
print(confusion_matrix(y_preds,y))
plot_conf_mat(y, y_preds)
```

Şekil 5.47. Ölçeklenmiş veri ile yapılan eğitim işlemi ve karmaşıklık matrisi oluşturmak için yazılan kodların ekran görüntüsü

```
[[ 858   44   50   38   10    0   21   53]
 [  23  810  111   15    6    0   12   41]
 [  11   13 1547   41   17    0   11   34]
 [  52   67  156 1281   32    2   84  106]
 [  14   28   73   36  444    1   35  118]
 [   0    1    1    1    0    8    0    0]
 [   7   13   35   38   11    1 1069   60]
 [   7   12   24    9   13    0   15 2380]]
```

Şekil 5.48. Ölçeklenmiş veri ile yapılan eğitim sonrası elde edilen karmaşıklık matrisi ekran görüntüsü



Şekil 5.49. Ölçeklenmiş veri ile yapılan eğitim sonrası elde edilen karmaşıklık matrisi ısı haritası ekran görüntüsü

Ölçeklenmiş veri ile yapılan eğitim sonra başarım skorlarını görüntüleyebilmek için yazılan kodların ekran görüntüsü Şekil 5.50’ de gösterilmiştir.

```
from sklearn.metrics import precision_score,recall_score,f1_score,ac
curacy_score
accuracy_score(y,y_preds)
0.8397
recall_score(y,y_preds,average="weighted")
0.8397
precision_score(y,y_preds,average="micro")
0.8397
f1_score(y,y_preds,average="micro")
0.8396999999999999
```

Şekil 5.50. Ölçeklenmiş veri ile yapılan eğitim sonucu skorları görüntülemek için yazılan kodların ekran görüntüsü

Ölçeklenmiş veri ile yapılan işlemler sonrasında başarım skoru 0.8397 olarak elde edilmiştir. Model yüksek skor almış olmasına karşın, Şekil 5.44'de görülen karmaşıklık matrisinde gerçek veriler ve tahmin edilen veriler arasındaki uyumun düşük olması nedeniyle, ölçeklendirilmiş veri üstünden eğitim gerçekleştirilerek tekrar Şekil 5.48' da görülen karmaşıklık matrisi ve Şekil 5.49' de görülen ısı haritası elde edilmiştir. Elde edilen tablolar incelendiğinde, ölçeklenmiş veri ile yapılan işlemdeki tahmin değerleri ve gerçek değerlerin örtüşme oranlarının daha yüksek olduğu gözlemlenmiştir. Bu da, makine öğrenmesi süreçlerinde veri ölçeklenmesinin gerekliliğini ortaya koymaktadır.

Bu noktaya kadar yapılan çalışmalardan elde edilen bulgular on bin satırlık veri seti üstünden ortaya çıkarılmıştır. Makine öğrenmesi model seçimi yol haritası incelendiğinde, yüz bin satır verinin üstünde bulunan veri setleri için öncelikle olasılıksal dereceli azalma yöntemi, çalışmadığı durumda ise çekirdek yaklaşımı yönteminin kullanılması önerilmektedir.

Bu doğrultuda, bölüm 4.1'de anlatılan sentetik veri seti üretim süreçleri tekrar işletilerek yüz bin on satırlık bir veri seti elde edilmiştir. Elde edilen veri seti ile olasılıksal dereceli azalma yöntemi ve çekirdek yaklaşımı yöntemleri ile eğitim süreçleri işletilerek model performansları değerlendirilmiştir.

5.2.4. Olasılıksal dereceli azalma sınıflandırıcısı (Stochastic gradient descent- Sgd) için geliştirilen kodlar ve sonuçları

Olasılıksal dereceli azalma sınıflandırıcısı ile çalışabilmek adına, Şekil 5.51'de görülen ilgili kütüphaneler tanımlanarak veri seti yüklenmiştir.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.linear_model import SGDClassifier
from sklearn.preprocessing import MinMaxScaler,StandardScaler
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.model_selection import RandomizedSearchCV,GridSearchCV
from sklearn.metrics import confusion_matrix,classification_report
from sklearn.metrics import precision_score,recall_score,f1_score
from sklearn.metrics import roc_curve,plot_confusion_matrix,plot_roc_cur
ve
bilgi = pd.read_csv("data100010.csv")
```

Şekil 5.51. Sgd sınıflandırıcısı için gerekli olan kütüphaneler ve veri setinin yüklenmesi için yazılan kodların ekran görüntüsü

Yüz bin on satırlık veri setinin yüklenerek ilk beş ve son beş satır örneği ekran görüntüsü Şekil 5.52'de gösterilmiştir.

	yukseklik	tasiyici	kullanim	durum	hasar	komsu	kamubina	sudogalgaz	gurultu	toz	titresim	yolmesafe	sonuc
0	30	1	6	1	1	5	0	0	39	3	415	90	7
1	57	5	1	0	2	80	0	1	69	2	315	40	2
2	45	4	6	1	3	45	0	0	43	2	338	15	7
3	27	4	2	1	3	15	0	1	75	2	290	5	4
4	3	5	6	1	2	65	1	1	60	1	435	75	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
100005	30	2	5	1	3	70	0	1	67	3	330	45	3
100006	60	1	3	1	3	25	0	0	44	2	469	100	7
100007	12	4	1	1	3	75	0	1	48	2	510	15	1
100008	36	5	1	1	1	45	1	0	79	4	538	10	7
100009	27	4	4	1	2	100	0	0	73	3	568	55	2

100010 rows × 13 columns

Şekil 5.52. 100010 satırlık veri setinin ilk beş ve son beş örneği ekran görüntüsü

Verilerin eğitime hazırlanabilmesi için, ölçeklendirme yapılarak eğitim ve test verileri olarak ayrıştırma işlemi yapılmıştır. Bu işlem için yazılan kodların ekran görüntüsü Şekil 5.53'de gösterilmiştir. Test verileri, tüm veri setinin yüzde 20'si olarak belirlenmiştir.

```
X = bilgi.drop("sonuc",axis=1)
y = bilgi["sonuc"]
X_egit,X_test,y_egit,y_test = train_test_split(X,y,test_size=0.2)
scaler = StandardScaler()
scaler.fit(X_egit)
train_set_scaled = scaler.transform(X_egit)
test_set_scaled = scaler.transform(X_test)
scaled_X = scaler.transform(X)
```

Şekil 5.53. Veri üstünde yapılan ölçeklendirme işlemi ve verilerin eğitim-test verisi olarak bölünmesi ekran görüntüsü

Verilerin ölçeklendirilip eğitim ve test verileri olarak ayrıştırılmasından sonra, SGD ile eğitim işlemi gerçekleştirilmiştir. Eğitim işleminde loss parametresinin değeri "hinge" olarak belirtilmiştir. Bu değer, SGD sınıflandırıcısının varsayılan değeri olmakla birlikte linear SVM modelinin kullanılacağını belirtmektedir. Bu modelin kullanılmasının amacı, bina verilerinin örnek sayısının arttırılmasıyla birlikte svm'in çok sayıda veride daha iyi eğitim performansı göstermesidir. SGD, bir makine öğrenmesi modeli değil, bir modelin eğitimi için kullanılan yöntemlerden birisidir (Pedregosa ve diğerleri, 2011). Veri setinin eğitimi işlemi için yazılan kodların ekran görüntüsü Şekil 5.54' de gösterilmiştir.

```
SGDC = SGDClassifier(loss="hinge", penalty="l2", max_iter=5)
SGDC.fit(X,y)
SGDClassifier(alpha=0.0001, average=False, class_weight=None,
              early_stopping=False, epsilon=0.1, eta0=0.0,
fit_intercept=True,
              l1_ratio=0.15, learning_rate='optimal', loss='hinge',
max_iter=5,
              n_iter_no_change=5, n_jobs=None, penalty='l2',
power_t=0.5,
              random_state=None, shuffle=True, tol=0.001,
              validation_fraction=0.1, verbose=0, warm_start=False)
```

Şekil 5.54. SGD Sınıflandırıcı ile eğitim işlemi kodları ekran görüntüsü

Eğitim işleminin sonuçlanması ile elde edilecek tahmin başarım skoru değerinin daha yüksek çıkması adına RandomizedSearchCV fonksiyonu ile parametre seçimi yapılarak, en yüksek başarım skoru aranmıştır.

```
from scipy.stats import randint,norm,uniform
param_dist = {"alpha": uniform(scale=0.01),
              "loss":["hinge","log"],
              "penalty": ["none", "l1", "l2"]}

sgd_clf = SGDClassifier()
sgd_random = RandomizedSearchCV(estimator = sgd_clf, param_distribut
ions = param_dist,n_iter=10)
sgd_random.fit(train_set_scaled, y_egit)

RandomizedSearchCV(cv=None, error_score=nan,
                   estimator=SGDClassifier(alpha=0.0001, average=False,
                                           class_weight=None,
                                           early_stopping=False, epsilon=0.1,
                                           eta0=0.0, fit_intercept=True,
                                           l1_ratio=0.15,
                                           learning_rate='optimal',
                                           loss='hinge', max_iter=1000,
                                           n_iter_no_change=5, n_jobs=None,
                                           penalty='l2', power_t=0.5,
                                           random_state=None, shuffle=True,
                                           tol=0.001, validation_fraction=0.1,
                                           verbose=0, warm_start=False),
                   iid='deprecated', n_iter=10, n_jobs=None,
                   param_distributions={'alpha':
<scipy.stats._distn_infrastructure.rv_frozen object at 0x7fb4c68238d0>,
                                        'loss': ['hinge', 'log'],
                                        'penalty': ['none', 'l1', 'l2']},
                   pre_dispatch='2*n_jobs', random_state=None, refit=True,
                   return_train_score=False, scoring=None, verbose=0)
```

Şekil 5.55. RandomizedSearchCV fonksyionu ile en uygun parametrelerin seçilerek başarım skorunu arttırmak adına yazılan kodların ekran görüntüsü

RandomizedSearchCV fonksiyonu, hiperparametre optimizasyonlarında rastgele olarak seçim yapmasından dolayı çalışma performansı yüksek bir optimizasyon tekniği olarak değerlendirilmektedir. Bununla birlikte tüm parametreleri seçmediğinden, en yüksek başarım skorunu elde edebilecek parametreleri seçme garantisi bulunmamaktadır. Bu nedenlerden dolayı tüm parametreleri tek tek deneyebilmek adına, çalışma performansı daha düşük de olsa Şekil Şekil 5.56' de görülen GridSearchCV fonksiyonu çalıştırılarak da optimizasyon sağlanmıştır. Hiperparametre optimizasyonu ile ilgili detaylar bölüm 5.2.1'de anlatılmıştır.

```
param_grid = {
               "alpha": [0.0001, 0.001, 0.01, 0.1, 1, 10, 100],
               "loss":["hinge","log"],
               "penalty": ["none", "l1", "l2"]}

sgd_clf = SGDClassifier()
sgd_grid = GridSearchCV(estimator = sgd_clf, param_grid=param_grid)
sgd_grid.fit(train_set_scaled, y_egit)
GridSearchCV(cv=None, error_score=nan,
             estimator=SGDClassifier(alpha=0.0001, average=False,
                                     class_weight=None, early_stopping=False,
                                     epsilon=0.1, eta0=0.0, fit_intercept=True,
                                     l1_ratio=0.15, learning_rate='optimal',
                                     loss='hinge', max_iter=1000,
                                     n_iter_no_change=5, n_jobs=None,
                                     penalty='l2', power_t=0.5,
                                     random_state=None, shuffle=True, tol=0.001,
                                     validation_fraction=0.1, verbose=0,
                                     warm_start=False),
             iid='deprecated', n_jobs=None,
             param_grid={'alpha': [0.0001, 0.001, 0.01, 0.1, 1, 10, 100],
                         'loss': ['hinge', 'log'],
                         'penalty': ['none', 'l1', 'l2']},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0)


sgd_random.best_score_,sgd_random.best_estimator_,sgd_random.best_params
_

(0.5427457387462841,
 SGDClassifier(alpha=0.0028626773138695017, average=False, class_weight=None,
               early_stopping=False, epsilon=0.1, eta0=0.0, fit_intercept=True,
               l1_ratio=0.15, learning_rate='optimal', loss='log',
max_iter=1000,
               n_iter_no_change=5, n_jobs=None, penalty='none', power_t=0.5,
               random_state=None, shuffle=True, tol=0.001,
               validation_fraction=0.1, verbose=0, warm_start=False),
 {'alpha': 0.0028626773138695017, 'loss': 'log', 'penalty': 'none'})

sgd_grid.best_score_,sgd_grid.best_estimator_,sgd_grid.best_params_

(0.5523822950979324,
 SGDClassifier(alpha=0.1, average=False, class_weight=None,
early_stopping=False,
               epsilon=0.1, eta0=0.0, fit_intercept=True, l1_ratio=0.15,
               learning_rate='optimal', loss='log', max_iter=1000,
               n_iter_no_change=5, n_jobs=None, penalty='none', power_t=0.5,
               random_state=None, shuffle=True, tol=0.001,
               validation_fraction=0.1, verbose=0, warm_start=False),
 {'alpha': 0.1, 'loss': 'log', 'penalty': 'none'})
```

Şekil 5.56. GridSearchCV fonksiyonu kullanılarak en iyi parametre seçimi için yazılan kodlar ve sonuçlarının ekran görüntüsü

Verilerin ölçeklendirilip, en iyi parametrelerin seçimi sonrasında yapılan eğitim ile elde edilen yıkım tekniği tahminleri ve veri setinde bulunan gerçek değerlerin örtüşme oranlarını görebilmek amacıyla Şekil 5.58'de görülen karmaşıklık matrisi ve Şekil 5.60'da görülen ısı haritası oluşturulmuştur.

```
model  = sgd_grid.best_estimator_
y_preds = model.predict(X)
print(confusion_matrix(y_preds,y))
```

Şekil 5.57. Karmaşıklık matrisi oluşturmak için yazılan kodların ekran görüntüsü

```
[[    0     0     0     0     0     0     0     0]
 [    0     0     0     0     0     0     0     0]
 [ 3698  1746   341     0   127     0   256     0]
 [ 5785  8009 19218 14711  5118   168 24759  8957]
 [    0     0     0     0     0     0     0     0]
 [    0     0     0     0     0     0     0     0]
 [    0     0     0     0     0     0     0     0]
 [   17    63    39    46   667     0  3223  3062]]
```

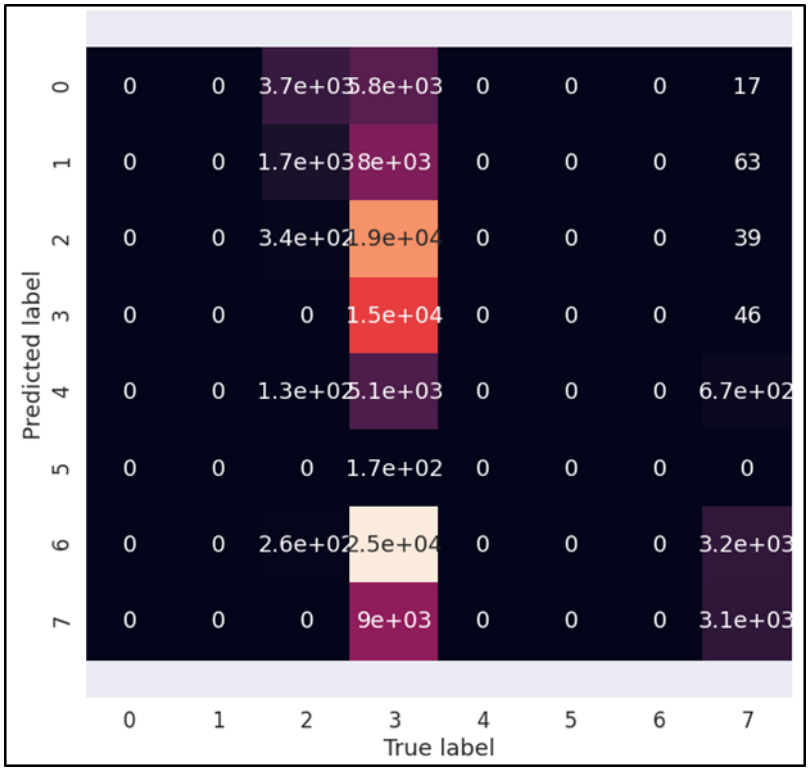
Şekil 5.58. Elde edilen karmaşıklık matrisi ekran görüntüsü

Şekil 5.57' da görülen karmaşıklık matrisinin ısı haritası da oluşturulmuştur. Isı haritası oluşturmak için yazılan kodların ekran görüntüsü Şekil 5.59' de gösterilmiştir.

```
sns.set(font_scale=1.5)
def plot_conf_mat(y_test, y_preds):
    fig, ax = plt.subplots(figsize=(10, 10))
    ax = sns.heatmap(confusion_matrix(y_test, y_preds),
                     annot=True,
                     cbar=False)
    plt.xlabel("True label")
    plt.ylabel("Predicted label")

    bottom, top = ax.get_ylim()
    ax.set_ylim(bottom + 0.5, top - 0.5)
plot_conf_mat(y, y_preds)
```

Şekil 5.59. Karmaşıklık matrisi ısı haritası oluşturmak için yazılan kodların ekran görüntüsü

Kodların çalıştırılmasının ardından elde edilen karmaşıklık matrisi ısı haritası ekran görüntüsü Şekil 5.60' de görülmektedir.



Şekil 5.60. Optimizasyon sonrası elde edilen karmaşıklık matrisi ısı haritası

```
from sklearn.metrics import precision_score,recall_score,f1_score,ac
curacy_score
accuracy_score(y,y_preds)
0.18112188781121888
recall_score(y,y_preds,average="weighted")
0.18112188781121888
precision_score(y,y_preds,average="micro")
0.18112188781121888
f1_score(y,y_preds,average="micro")
0.18112188781121888
```

Şekil 5.61. Optimizasyon sonrası sgd sınıflandırıcı ile elde edilen başarım skorları

Şekil 5.61 incelendiğinde optimizasyon sonrası yapılan eğitim işlemi sonrasında elde edilen başarım skoru 0.1811 olmuştur ve düşük bir başarım skoru olarak değerlendirilmiştir. Şekil 5.60' daki karmaşıklık matrisi ısı haritası incelendiğinde de, düşük başarım skorunun etkileri tahmin ve gerçek değerler arasındaki örtüşme düzeylerinden gözlemlenebilmektedir.

Üretilen veri seti temelinde olasılıksal dereceli azalma sınıflandırıcısı ile elde edilen bulguların, yıkım teknikleri seçiminde kullanılamayacağı görüldüğünden, aynı veri seti önerilen son yöntem olan çekirdek yaklaşımı yöntemi denenmiştir.

5.2.5. Çekirdek yaklaşımı (Kernel approximation) için geliştirilen kodlar ve sonuçları

Çekirdek yaklaşımı, destek vektör sınıflandırıcısı gibi çekirdek eşleşmesi kullanan modellerin büyük veriler üzerinden örtük işlemler yapması yerine açık eşleşme yaparak daha verimli sonuçlar elde edilmesini amaçlamaktadır.

Bu yaklaşımın kullanılmasında gerekli olan kütüphaneler Şekil 5.62 ve Şekil 5.63' deki ekran görüntülerinde gösterilmiştir.

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from time import time
```

Şekil 5.62. Çekirdek yaklaşımı standart kütüphaneleri kullanmak için yazılan kodların ekran görüntüsü

```
from sklearn import svm, pipeline
from sklearn.kernel_approximation import (RBFSampler,Nystroem)
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler as Scaler
import seaborn as sns
```

Şekil 5.63. Sınıflama modellerinin ve performans metrikleri için kullanılacak kütüphanelerin eklenmesi için yazılan kodların ekran görüntüsü

Dönüştürücüler, kompozit bir model oluşturmak için kullanılmaktadırlar. Bu model, içerisinde sınıflandırıcı ya da regresör tipinde tahminden edicileri barındırmaktadır. Bu kompozit modelin elde edilmesinde pipeline sıklıkla kullanılmaktadır. Bir başka ifadeyle pipeline birden fazla tahmin ediciyi bir seferde zincirleme çalıştırmak için kullanılmaktadır. Çekirdek yaklaşımı hakkında çok fazla deneysel çalışma olmadığından mevcut çalışmanın kesin çekirdek yöntemleriyle karşılaştırılması önerilmektedir.

```
from google.colab import drive
drive.mount('/content/drive')
bilgiDf = pd.read_csv("data100010.csv")
```

Şekil 5.64. Veri setinin yüklenmesi için kullanılan kodların ekran görüntüsü

Veri setinin yüklenmesinin ardından etiketlerin veri setinden ayrıştırılıp ölçeklendirilerek eğitim ve test verileri elde edilmiştir. Bu işlem için yazılan kodlar Şekil 5.65' de gösterilmiştir.

```
X = bilgiDf.drop("sonuc",axis=1)
y = bilgiDf["sonuc"]
egitimDf_X,testDf_X,egitimDf_y,testDf_y = train_test_split(X,y,test_s
ize=0.2)
scaler = Scaler()
scaler.fit(X)
egitimDf_X_scaled = scaler.transform(egitimDf_X)
testDf_X_scaled = scaler.transform(testDf_X)

kernel_svm = svm.SVC(gamma=.2)
linear_svm = svm.LinearSVC()
```

Şekil 5.65. Etiketlerin veri setinden ayrıştırılıp ölçeklendirilerek eğitim ve test verisinin elde edilmesi için yazılan kodların ekran görüntüsü

Verilerin ölçeklendirilerek eğitim ve test verilerinin edilmesinin ardından kompozit bir model oluşturulmuştur. Şekil 5.66'de görülen kodlar ile geliştirilen bu model ile zincirleme Şekilde tahmin ediciler belirlenmiştir.

```
feature_map_fourier = RBFSampler(gamma=.2, random_state=1)
feature_map_nystroem = Nystroem(gamma=.2, random_state=1)
fourier_approx_svm = pipeline.Pipeline([("feature_map", feature_map_f
ourier),
                                        ("svm", svm.LinearSVC())])
nystroem_approx_svm = pipeline.Pipeline([("feature_map", feature_map_
nystroem),
                                         ("svm", svm.LinearSVC())])
```

Şekil 5.66. Kernel approximation ile linear svc üzerinden pipeline ile kompozit bir model oluşturmak için yazılan kodların ekran görüntüsü

Pipeline kompozit bir model oluşturulmasının ardından, Linear Svc ve çekirdek yaklaşımı kullanılarak Şekil 5.67'da kodları görülen eğitim işlemi gerçekleştirilmiştir

```
kernel_svm_time = time()
kernel_svm.fit(egitimDf_X_scaled,egitimDf_y)
kernel_svm_score = kernel_svm.score(testDf_X_scaled,testDf_y)
kernel_svm_time = time() - kernel_svm_time


linear_svm_time = time()
linear_svm.fit(egitimDf_X_scaled, egitimDf_y)
linear_svm_score = linear_svm.score(testDf_X_scaled, testDf_y)
linear_svm_time = time() - linear_svm_time
```

Şekil 5.67. Linear svc ve çekirdek yaklaşımıyla verilerin amacıyla yazılan kodların ekran görüntüsü

feature_map__n_components ile hesaplanan özellik sayısının ve kullanılan özellik sayısı değiştirilerek oluşturulan kompozit modellerin başarımı arttırılmaya çalışılmıştır.

```
sample_sizes = 100 * np.arange(1, 10)
fourier_scores = []
nystroem_scores = []
fourier_times = []
nystroem_times = []
feature_map_comp = []
adim = 0
for D in sample_sizes:
    feature_map_comp.append(D)
    fourier_approx_svm.set_params(feature_map__n_components=D)
    nystroem_approx_svm.set_params(feature_map__n_components=D)
    start = time()
    nystroem_approx_svm.fit(egitimDf_X_scaled, egitimDf_y)
    nystroem_times.append(time() - start)
    start = time()
    fourier_approx_svm.fit(egitimDf_X_scaled, egitimDf_y)
    fourier_times.append(time() - start)
    fourier_score = fourier_approx_svm.score(testDf_X_scaled, testDf_y)
    nystroem_score = nystroem_approx_svm.score(testDf_X_scaled, testDf_
y)
    nystroem_scores.append(nystroem_score)
    fourier_scores.append(fourier_score)
```

Şekil 5.68. Kompozit modellerin başarımının arttırımı için yazılan kodların ekran görüntüsü

Başarım arttırma işleminden sonra elde edilen skorlar yazdırılmıştır.

```
print(nystroem_scores)
print(fourier_scores)
print(feature_map_comp)


[0.5357, 0.58475, 0.6108, 0.6287, 0.6405, 0.64905, 0.65705, 0.6634,
0.6681]
[0.4121, 0.48145, 0.53065, 0.55065, 0.57195, 0.59055, 0.61115, 0.62095,
0.61815]
[100, 200, 300, 400, 500, 600, 700, 800, 900]
```

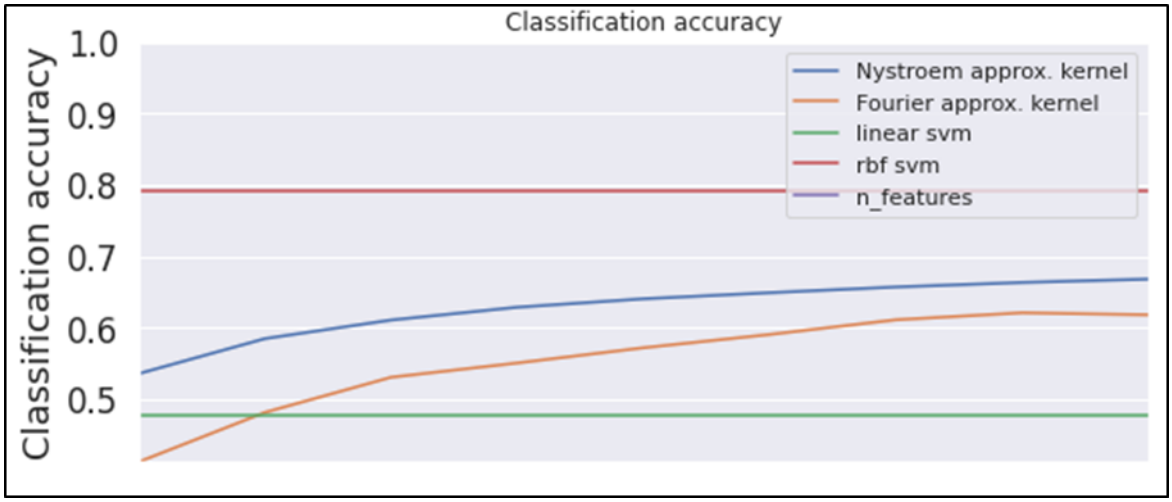
Şekil 5.69. Başarım arttırımından sonra elde edilen skorlar

Şekil 5.69' da görülen elde edilen skorlar incelendiğinde, nystorem yöntemi ile alınan skor 0.6681, fourier yöntemi ile alınan skor 0.6181 olmuştur. Bu başarım skorlarının yıkım teknikleri seçimi konusunda karar desteği vermek için yeterli olmadığı kararı verilmiştir.

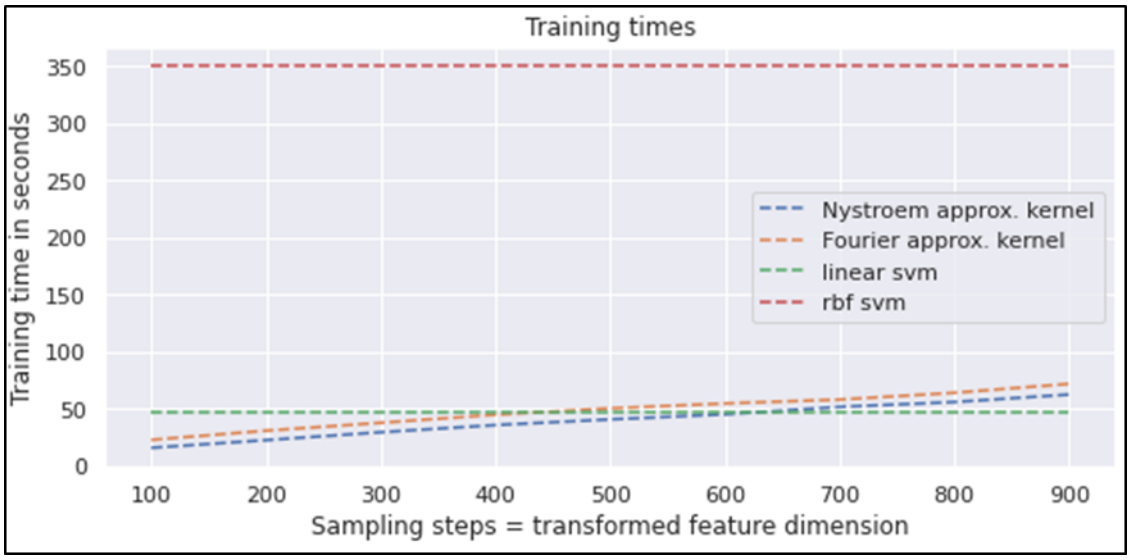
Oluşturulmuş kompozit modeller ile bu modellerin doğal hallerinin eğitim sürecinde zaman ve skor bakımından karşılaştırılması yapılmıştır.

```
plt.figure(figsize=(16, 4))
accuracy = plt.subplot(121)
sns.set(font_scale=1)
# second y axis for timings
timescale = plt.subplot(122)
accuracy.plot(sample_sizes, nystroem_scores, label="Nystroem approx.
kernel")
timescale.plot(sample_sizes, nystroem_times, '--
',label='Nystroem approx. kernel')

accuracy.plot(sample_sizes, fourier_scores, label="Fourier approx. ke
rnel")
timescale.plot(sample_sizes, fourier_times, '--',
               label='Fourier approx. kernel')
# horizontal lines for exact rbf and linear kernels:
accuracy.plot([sample_sizes[0], sample_sizes[-1]],
              [linear_svm_score, linear_svm_score], label="linear svm
")
timescale.plot([sample_sizes[0], sample_sizes[-1]],
               [linear_svm_time, linear_svm_time], '--
', label='linear svm')

accuracy.plot([sample_sizes[0], sample_sizes[-1]],
              [kernel_svm_score, kernel_svm_score], label="rbf svm")
timescale.plot([sample_sizes[0], sample_sizes[-1]],
               [kernel_svm_time, kernel_svm_time], '--
', label='rbf svm')
# vertical line for dataset dimensionality = 64
accuracy.plot([64, 64], [0.7, 1], label="n_features")
# legends and labels
accuracy.set_title("Classification accuracy")
timescale.set_title("Training times")
accuracy.set_xlim(sample_sizes[0], sample_sizes[-1])
accuracy.set_xticks(())
accuracy.set_ylim(np.min(fourier_scores), 1)
timescale.set_xlabel("Sampling steps = transformed feature dimension"
)
accuracy.set_ylabel("Classification accuracy")
timescale.set_ylabel("Training time in seconds")
accuracy.legend(loc='best')
timescale.legend(loc='best')
plt.tight_layout()
plt.show()
```

Şekil 5.70. Kompozit modellerin eğitim süreçlerinde zaman ve skor açısından karşılaştırmaları amacıyla yazılan kodların ekran görüntüsü



Şekil 5.71. Sınıflandırma doğruluk grafiği ekran görüntüsü



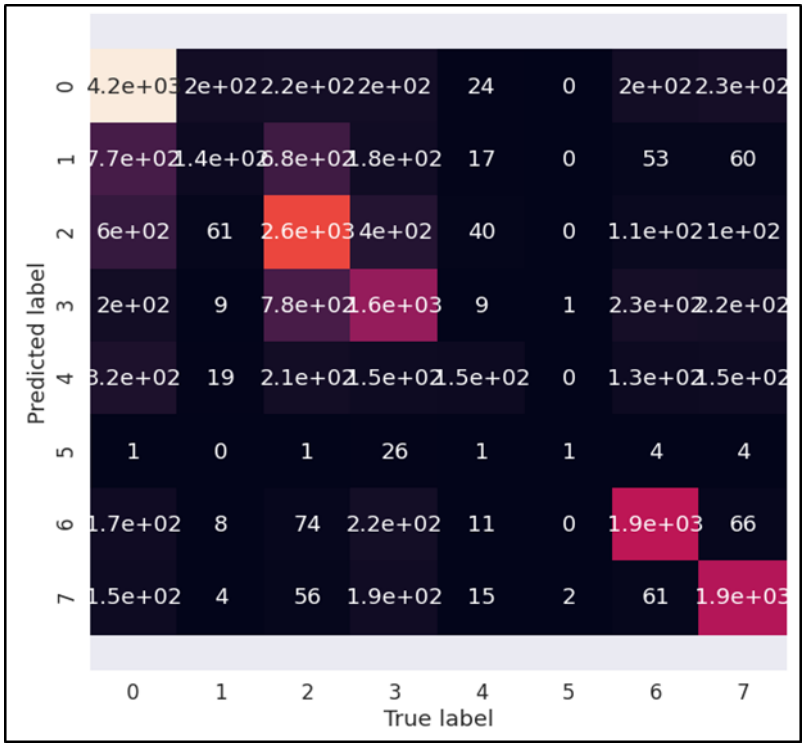
Şekil 5.72. Eğitim süreleri grafiği ekran görüntüsü

Şekil 5.71 ve Şekil 5.72'de gösterilen grafikler incelendiğinde çekirdek yaklaşımından kullanılan Nystroem ve Fourier yöntemlerinin, Nystroem yöntemi, temelinde rbf kullanıyor olmasına rağmen svm modeline göre skor anlamında başarılı olamamıştır. Ancak svm 'in süresinin büyüklüğü dikkate değer olarak gözlemlenmektedir.

Elde edilen mevcut kompozit modellerin başarımı, veri setindeki gerçek sonuçlarla karşılaştırılmıştır. Bu karşılaştırma için karmaşıklık matrisi ısı haritaları kullanılmıştır. Fourier yöntemi kullanılarak elde edilen başarım karşılaştırmaları amacıyla yazılan kodların ekran görüntüsü Şekil 5.73' de gösterilmiştir.

```
fourier_approx_svm.set_params(feature_map__n_components=900)
y_preds =  fourier_approx_svm.predict(testDf_X_scaled)

from sklearn.metrics import confusion_matrix
sns.set(font_scale=1.5)

def plot_conf_mat(y_test, y_preds):
    fig, ax = plt.subplots(figsize=(10, 10))
    ax = sns.heatmap(confusion_matrix(y_test, y_preds),
                     annot=True,
                     cbar=False)
    plt.xlabel("True label")
    plt.ylabel("Predicted label")

    bottom, top = ax.get_ylim()
    ax.set_ylim(bottom + 0.5, top - 0.5)

plot_conf_mat(testDf_y, y_preds)
```

Şekil 5.73. Fourier yöntemi kullanılarak elde edilen başarım karşılaştırmaları amacıyla yazılan kodların ekran görüntüsü

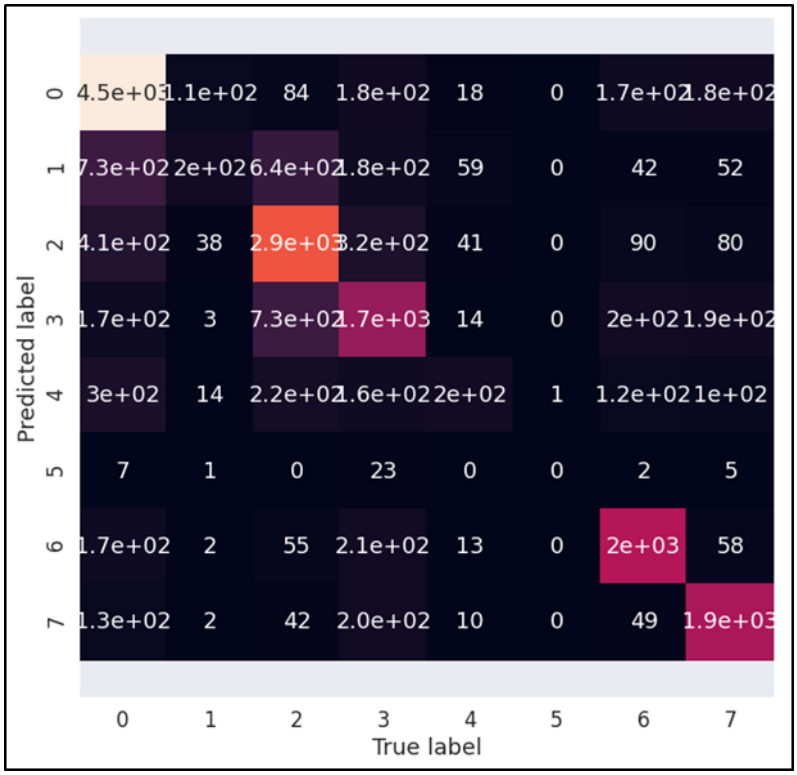


Şekil 5.74. Fourier yöntemi kullanılarak elde edilen başarım karşılaştırmaları ısı haritası

Nystrom yöntemi kullanılarak elde edilen başarım karşılaştırmaları amacıyla yazılan kodların ekran görüntüsü Şekil 5.74' de gösterilmiştir.

```
nystroem_approx_svm.set_params(feature_map__n_components=900)
y_preds =  nystroem_approx_svm.predict(testDf_X_scaled)

import seaborn as sns
from sklearn.metrics import confusion_matrix
sns.set(font_scale=1.5)

def plot_conf_mat(y_test, y_preds):
    fig, ax = plt.subplots(figsize=(10, 10))
    ax = sns.heatmap(confusion_matrix(y_test, y_preds),
                     annot=True,
                     cbar=False)
    plt.xlabel("True label")
    plt.ylabel("Predicted label")

    bottom, top = ax.get_ylim()
    ax.set_ylim(bottom + 0.5, top - 0.5)

plot_conf_mat(testDf_y, y_preds)
```

Şekil 5.75. Nystrom yöntemi kullanılarak elde edilen başarım karşılaştırmaları amacıyla yazılan kodların ekran görüntüsü



Şekil 5.76. Nystrom yöntemi kullanılarak elde edilen başarım karşılaştırmaları ısı haritası

Çekirdek yaklaşımı hesaplama yöntemlerinden olan fourier ve nystrom yöntemlerinden elde edilen sonuçlara ilişkin Şekil 5.74 ve Şekil 5.76' deki karmaşıklık matrisi ısı haritaları incelendiğinde, veri setinde etiketlenmiş gerçek değerler ile tahmin edilen değerler arasındaki örtüşme seviyesinin düşük olduğu gözlemlenmiştir. Bunun sebebi, nystrom yöntemi yapılan eğitim sonucunda başarım skoru olarak 0.6681, fourier yöntemi ile yapılan eğitim sonucunda elde edilen başarım skorunun 0.6181 olduğu söylenebilir.

Veri seti üstünde yapılan eğitim işlemlerinden elde edilen bulgulara göre, veri sayısının on bin satır olduğunda, k en yakın komşu algoritması ile en iyi sonucun alındığı gözlemlenmiştir. Elde edilen bu bulgu ile, k en yakın komşu modeli kayıt edilerek web tabanlı bir yazılım ile entegre edilmiştir.

5.3. Web Tabanlı Yapay Zeka Destekli Yıkım Karar Destek Uygulaması

Yıkım uzmanları tarafından mobil, masaüstü, tablet farketmeksizin platform bağımsız olarak veri girişi yapılarak yıkım kararı önerileri yapılabilmesi amacıyla bir web tabanlı yapay zeka destekli yıkım karar destek uygulaması geliştirilmiştir. Bu bölümde uygulama geliştirme aşamaları belirtilmiştir.

5.3.1. Web tabanlı uygulama temelinde kullanılacak makine öğrenmesi modeli

Bölüm 5.2' de, üretilen sentetik veri setinin içinde bulunan verilerin niteliğine uygun makine öğrenmesi modelleri ve yöntemleri incelenmiştir. Elde edilen bulgular ışığında, veri setindeki verilerle yapılan eğitimler sonucunda en yüksek tahmin başarım skorunun k en yakın komşu algoritması ile elde edildiği gözlemlenmiştir. Bu noktada geliştirilen web uygulamasının karar destek temelinde k en yakın komşu algoritması ile elde edilen model kullanılmıştır.

5.3.2. Web tabanlı uygulama geliştirmede kullanılan teknolojiler

Bu bölümde web tabanlı uygulama geliştirilirken kullanılan teknolojiler ve yazılım dilleri ele alınmıştır.

Uygulama ara yüzünde kullanılan diller

Html

Uygulama ara yüzünde temel olarak HTML (Hyper Text Markup Language) kullanılmıştır.

Html dili, web' in işaretleme dilinin her zaman html olmasından dolayıdır. Html, sistematik olarak bilimsel dokümanların geliştirilmesi için kullanılsa da, yıllar içerisinde birçok doküman tipi için kullanılabilir hale gelmiştir ve web üzerinde kullanılmaktadır (Hickson ve Hyatt, 2011). Bu çerçeveden bakıldığında web için ara yüz geliştirme standardının html olduğu söylenebilir. Html ile bir web ara yüzünün temel yapısı oluşturulmaktadır.

Html'in gelişimi içinde, Html 5 sürümü ile birlikte platform bağımsız ve vektörel grafikler gibi açık biçimli dosyaları desteklemektedir. Bununla birlikte, geliştiriciler de eklentilere gerek duymaksızın zengin içerikli web ara yüzleri geliştirebilmektedirler (Vaughan-Nichols, 2010).

Css

Html ile oluşturulan ara yüz çatısının renklendirilmesi, boyutlandırılması, yazı tiplerinin belirlenmesi, içeriklerin konumlandırılması için kullanılan dile Css (Cascading style sheets) denilmektedir.

İlk bakışta css basit bir dil olarak görünmekle birlikte, yazım kuralları açısından bakıldığında gerçekten basit bir dildir. Temel anlamda stil sayfaları, stil kurallarından oluşan bir bütündür. Html'deki elementler için her stil kuralının bir seçicisi vardır ve elementlerin bu stilleri uygulamasını sağlarlar (Geneves, Layaida ve Quint, 2012).

Uygulama arayüzünün tasarımında duyarlı tasarım teknikleri kullanılmıştır. Duyarlı tasarımlar, uygulamanın kullanıldığı cihaz farketmeksizin her ekran boyutunda sorunsuz kullanılabilmesini sağlamaktadır. Duyarlı tasarım geliştirmek için de css kuralları ve javascript kullanılmıştır.

Javascript ve Jquery

Javascript, web uygulamalarında kullanıcı etkileşimi, dinamik içerikler, animasyonlar oluşturmak için kullanılan bir dildir.

Kullanıcıların html dokümanları içerisinde doğrudan yazılabilmektedir. Javascript yazım kuralları, Java programlama diline benzemektedir. Javascript ile olay tabanlı programlama yapılabilmekle birlikte sınıf yazmadan nesneler oluşturmak da mümkündür (Goodman, 2007: 17). Javascript'in bu imkanları sayesinde her html elementine erişilebilir ve elementler üstünde istenilen değişiklikler yapılabilmektedir.

Her ne kadar javascript yazımı kolay da olsa, sık kullanılan işlemlerin daha hızlı gerçekleştirilebilmesi için geliştirilen kütüphane, Jquery kütüphanesidir. Jquery kütüphanesi içerisinde javascript diliyle geliştirilmiş fonksiyonlar bulunmaktadır ve sık yapılan script işlemlerinin hızlı bir Şekilde yapılması sağlanmaktadır. Jquery ile uygulama geliştirmede kod geliştirme hızlı da olsa, yapılan işlemler doğrudan javascript ile yazıldığında daha performanslı çalışabilmektedir. Bunun nedeni, jquery kütüphanesi içerisindeki fonksiyonların kullanılabilmesi için öncelikle kütüphanenin yüklenmesi, sonrasında fonksiyonların çağırımlarının yapılmasıdır.

Web tabanlı uygulamada kullanılan programlama dili

Web tabanlı uygulamanın arka planında çalışarak veri işlemlerini yapan programlama dili olarak python programlama dili kullanılmıştır.

Python programlama dili, üst seviye, nesne yönelimli ve açık kaynaklı bir programlama dili olarak hızlı program geliştirmek için tasarlanmıştır (Lutz, 2001: 1). Bilgisayarlar yalnızca makine dilinden anlamaktadırlar. Makine dili kullanarak bilgisayar programları geliştirmek çok zor olduğundan, programlama dilleri geliştirilmiştir. Her programlama dilinin makine diline dönüştürülmesini sağlayan derleyicileri vardır, python programlama dili içinse bu işlen python yorumlayıcısı tarafından yapılmaktadır (Zelle, 2004: 9).

Python programlama dilinin hızlı geliştirmeye imkan sağlayan yapısı, bölüm 5'de anlatılan makine dili kütüphanelerinin de bu dil ile geliştirilmesinden dolayı, web tabanlı uygulama geliştirilirken de python programlama dili kullanılmıştır. Bu sayede, makine öğrenmesi

algoritmalarından elde edilen eğitilmiş verilerin yine aynı dil ile kayıt edilerek web tabanlı uygulamada kullanılabilir hale getirilmesinde kolaylaştırıcı bir unsur olmuştur.

Web tabanlı uygulama, python programlama dili ile django çatısı kullanılarak geliştirilmiştir. Django çatısı, bünyesinde web uygulamaları geliştirmek için kullanılan bileşenler bulundurmaktadır.

Django çatısı kullanılarak üst seviye web uygulamaları az sayıda satır kod yazılarak gerçekleştirilebilir. Django kullanımı basit, etkili ve esnektir. Bu nedenlerle python programlama dili ile web uygulamaları geliştirmede oldukça etkin bir Şekilde kullanılmaktadır. (Forcier, Bissex, ve Chun, 2008: 6).

Veritabanı

Kaydedilen yıkım teknikleri bilgilerini saklamak adına veritabanı olarak Sqlite veritabanı kullanılmıştır.

Sqlite, ilişkisel bir veritabanı yönetim sistemidir. Her ne kadar isminde “lite” da geçse, bu kelime ile anlatılmak istenen özelliklerinin azlığı değil, hafif gereksinimlerinin olmasındandır. Sqlite, kurulum için bir sunucu gerektirmezken tek başına bir dosya olarak saklanabilmektedir. Kullanımı içinse herhangi bir konfigürasyon yapmaya gerek bulunmazken aynı zamanda çalışırken de düşük sistem kaynakları kullanmaktadır (Kreibich, 2010: 1-2).

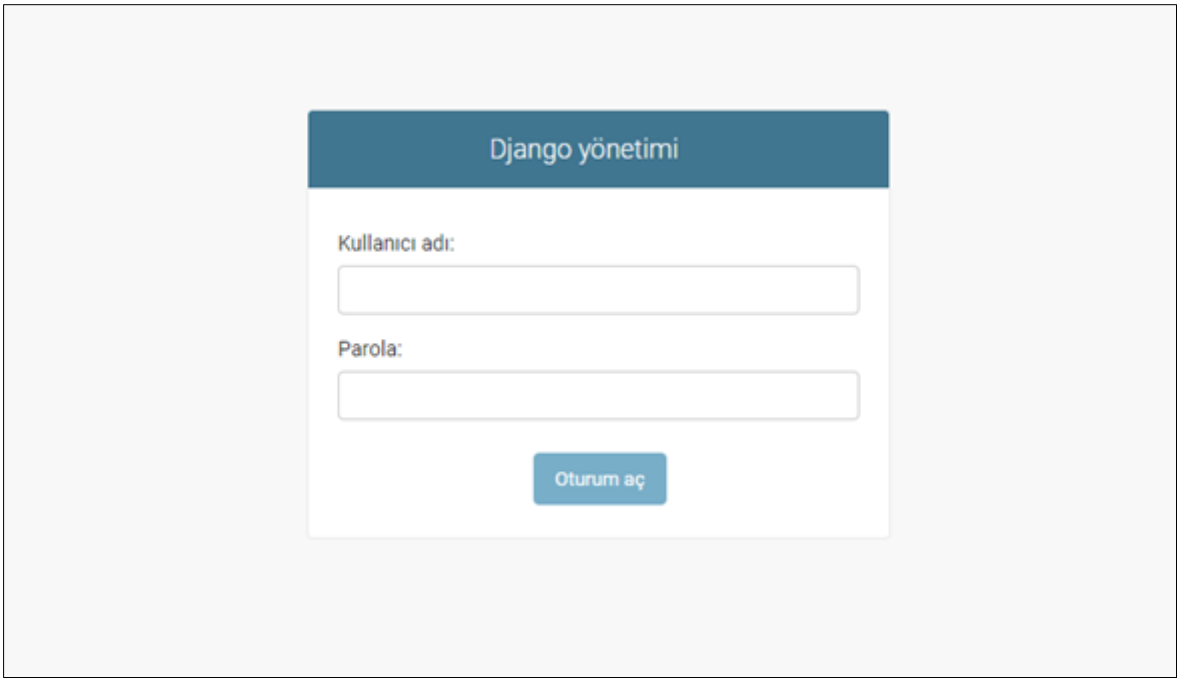
5.3.3. Veri tabanına yeni yıkım verileri giriş yöntemi

Makine öğrenmesi süreçlerinden geçirilerek elde edilen eğitilmiş veriler, oluşturulan veri tabanına kaydedilmiştir. Bu veriler, regülasyonlara ve uzman görüşlerine dayalı görüşlerle sentetik olarak üretilip, eğitim süreçlerinde kullanılmıştır. Tasarlanan web tabanlı yazılım ile bundan sonraki süreçlerde de veri tabanının yeni verilerle beslenmesinin sağlanabilmesi açısından, yıkım uzmanlarının veri girişi yapabilmesi için tamamen django çatısı ile geliştirilmiş bir kontrol paneli tasarlanmıştır. Bu panel ile yıkım uzmanları veri girişi yapabilmek için üyelik işlemlerini tamamladıktan sonra, yeni veriler ve bu verilere sahip yapının başarıyla yıkılması için gerekli tekniği seçerek veri tabanına katkıda

bulunabileceklerdir. Uzmanların veri tabanına ekleme yapabilmeleri için kontrol panelinden her uzman için kullanıcı ekleme işlemi yapılmaktadır. Kullanıcı ekleme için tasarlanan panelin ekran görüntüsü Şekil 5.77'deki gibidir.

Şekil 5.77. Yıkım bilgi sistemi uzman kayıt ekranı görüntüsü

Yıkım uzmanları kullanıcı olarak kaydedildikten sonra sisteme giriş için giriş ekranını kullanmaktadırlar. Giriş ekranı için yapılan tasarımın ekran görüntüsü Şekil 5.78'daki gibidir.

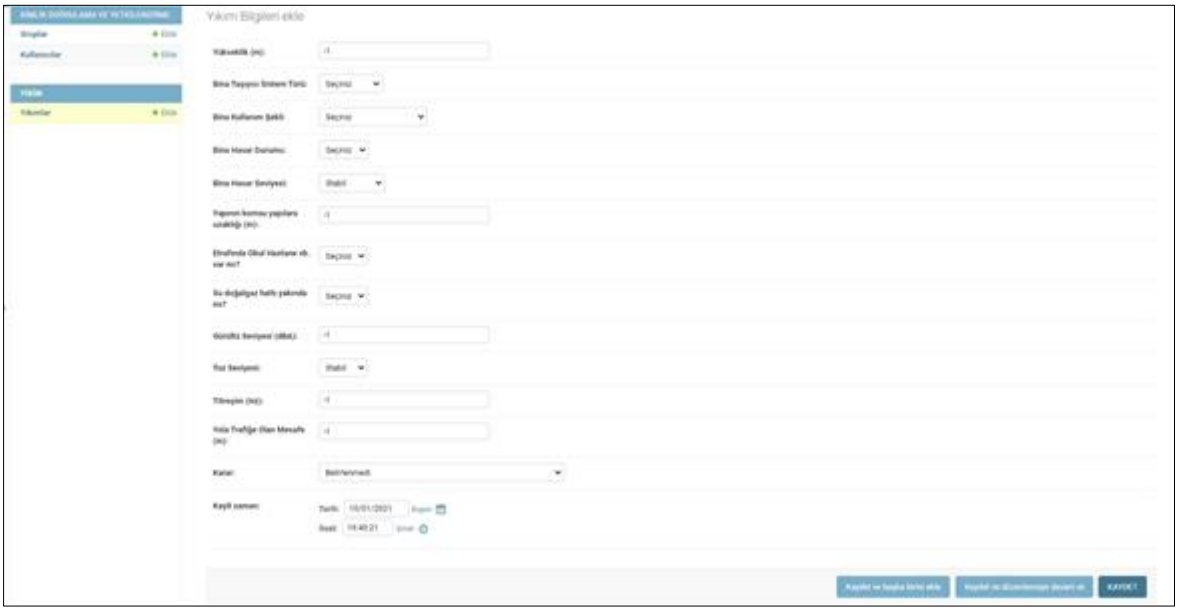


Şekil 5.78. Yıkım uzmanı kullanıcısı giriş ekranı

Veri tabanına kullanıcı olarak kaydedilen yıkım uzmanları, sisteme giriş yaptıktan sonra kendi belirleyeceği bina verileri ile yıkım kararı seçimi yaparak veri tabanına kayıt

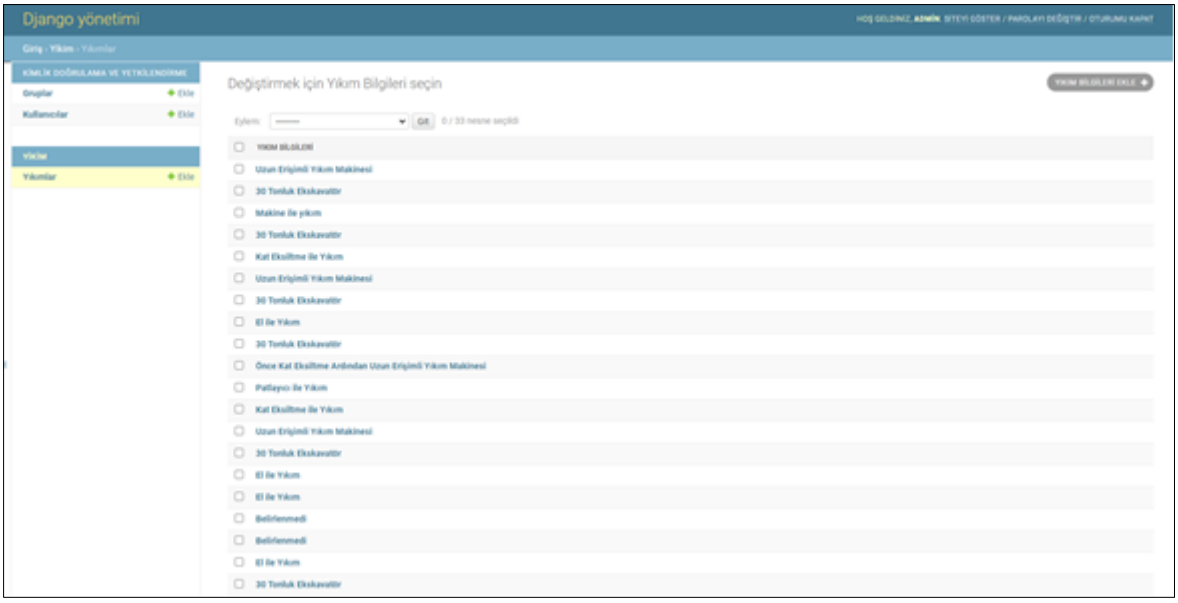
edebilmektedirler. Bu sistem ile, yıkım uzmanlarının görüşleri veri tabanına yeni veri olarak eklenerek veri setinin güncellenerek yeniden eğitim süreçlerine dahil edilmesi sağlanabilmektedir. Zaman içerisinde gelişen veri seti ile yeniden eğitim süreçleri işletildiğinde, başarım skorları daha yüksek tahminler elde edilebileceği düşünülmektedir.

Sisteme kayıt edilen yıkım uzmanlarının veri girişi yaptığı kontrol paneli ekran görüntüsü Şekil 5.79'de gösterilmektedir.



Şekil 5.79. Yıkım uzmanları tarafından yeni yıkım verisi ekleme paneli ekran görüntüsü

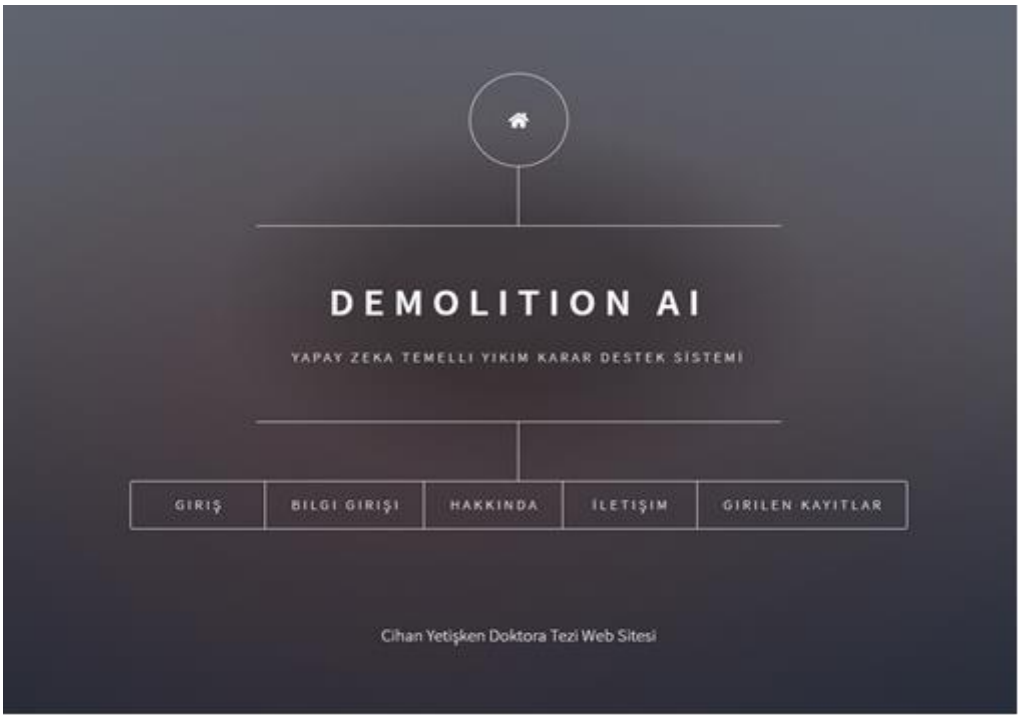
Veri tabanına kayıtlı yıkım uzmanları aynı zamanda veri tabanında daha önce kaydedilen yıkım verileri kararları üstünde de değişiklik yapabilmektedirler. Bununla birlikte veri tabanında bulunan kararların farklı uzmanlar tarafından değerlendirilip güncellenmesiyle doğruluğu arttırılmış kararların veri tabanında kayıt altına alınarak güncel tutulması hedeflenmiştir. Yıkım uzmanlarının veri tabanında bulunan önceki kayıtları güncelleyebileceği panelin ekran görüntüsü Şekil 5.80' de görülmektedir.



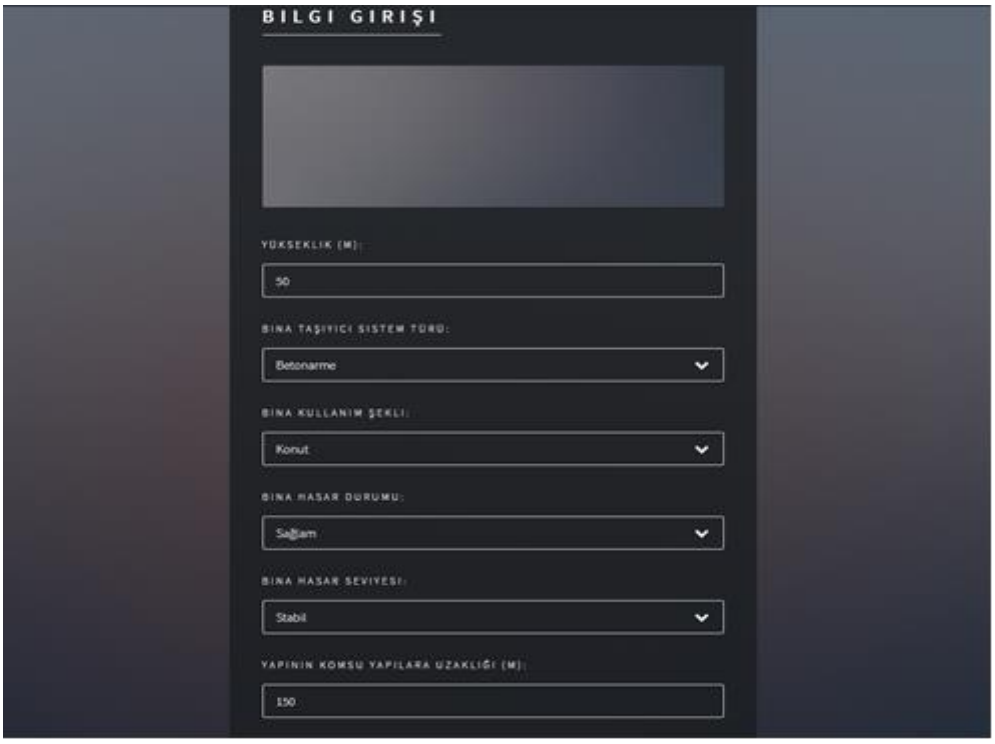
Şekil 5.80. Yıkım uzmanları tarafından güncellenebilen veri tabanındaki kayıtların ekran görüntüsü

5.3.4. Eğitilmiş verilerden karar desteği almak üzere geliştirilen ara yüz

Yıkım kararı verilen yapının özelliklerinin girilerek, yıkım tekniği karar desteği almak üzere geliştirilen web ara yüzünün ana sayfası ekran görüntüsü Şekil 5.81'de, verilerin uzmanlar tarafından girilerek, k en yakın komşu algoritması ile eğitilmiş verilerden tahmin yapıldıktan sonra yıkım kararının açıklandığı ekranların görüntüsü ise Şekil 5.82'de gösterilmiştir.



Şekil 5.81. Yıkım bilgi sistemi ana sayfası ekran görüntüsü



Şekil 5.82. Yıkımı yapılacak yapının özelliklerinin girilmesi ve yıkım kararı verilmesi için tasarlanan sayfanın ekran görüntüsü

Şekil 5.82'de görülmekte olan ekrandan yapının özellikleri girilerek gönder butonu ile yıkım kararının görüntülenmesi sağlanmaktadır.

6. SONUÇ VE ÖNERİLER

Yapı üretimi yanında çeşitli nedenlerle ömrünü tamamlayan yapıların yıkımı ve ortadan kaldırılması da onların yaşam döngüsü içerisinde gerçekleşen çok temel bir olgu olup gerçekleşen uygulamalar çerçevesinde çok sayıda hatalı uygulamalar olduğu değerlendirilmektedir. Bu kapsam üzerine kurgulanan bu tezin ana amacı, akademik olarak da sınırlı çalışmanın yer aldığı alandaki bu soruna destek oluşturabilecek yapı yıkım tekniği seçiminde yıkım uzmanlarına karar desteği veren makine öğrenmesi temelli bir sistemin geliştirilmesi olmuştur.

Cables ve arkadaşlarının 2016 yılında yaptığı çalışmaya göre, birden çok kriter içeren problemlerde, çok kriterli karar verme analizlerini kullanmak, karar vericilere yardımcı olmaktadır. Çok kriterli karar verme analiz yöntemleri, probleme sistematik bir Şekilde

yaklaşarak sonuca giden yolda karar vericilerin işlerini kolaylaştırmaktadır. Bina yıkımlarında kullanılan yıkım teknikleri seçimi birçok parametrenin aynı anda değerlendirilmesiyle karar verilen çok kriterli bir karar verme problemidir. Bu noktadan hareketle araştırma kapsamında yapı yıkım tekniği seçiminde önemli olan faktörler araştırılmıştır. Yapılan detaylı literatür taramasında ve uzman görüşmeleri sonucunda yapılara ait yükseklik, konum, alt yapı durumu, izin verilen gürültü düzeyi gibi yıkım tekniği kararına etki eden bir çok faktör arasında en önemli kriterin yükseklik olduğu görülmektedir. Uygulamada önem derecesi en yüksek kriter olan bu kriter temel alınarak diğer faktörlerin değerlendirilmesiyle birlikte yıkım tekniği kararı verilmektedir. Bu kararın verilmesinde, uzmanların geçmiş tecrübelerinden faydalanılması önemli görülmektedir (Şahmaran ve diğerleri, 2015).

Uygulanmakta olan bu mevcut durum, tezin geliştirilmesinde ilham kaynağı olmuş ve bu sürecin bilişim teknolojileri ile desteklenerek yürütülebileceği düşünülmüştür. Dixit ve arkadaşlarının 2020 yılında yaptığı çalışmaya göre, makine öğrenmesi teknikleri, gerçek zamanlı karar verebilmek için geliştirilmiş tekniklerdir. Bu noktadan bakıldığında, makine öğrenmesi tekniklerinin yapı yıkım tekniği karar desteğinde kullanılmasının önemli potansiyeli olduğu değerlendirilmiş ve bu temelde bir sistem önerisi geliştirilmiştir. Bu sistemin geliştirilme aşamalarından ilki ve en önemlisi olan veri seti hazırlığında başlangıç noktası olarak literatürden hareket edilmiştir. Yapılan literatür araştırmalarında, yıkım tekniği seçiminde kullanılabilecek nitelikte bir veri setine ulaşılmasının mümkün olmadığı görülmüş örnek bir çalışmaya rastlanamamıştır. Bu noktadan hareketle, uzman görüşleri ve ülkemizde uygulanan yönetmeliklerden beslenen sentetik bir veri seti oluşturulması kararı verilmiştir. Sentetik veri setinin niteliği ve çıktıları hem yönetmelikler hem de uzman görüşleri ile analiz edilmiş ve geçerliliğine dayalı olarak kullanılabilirlik kararı verilmiştir.

Veri seti oluşturulması aşamasında, yıkım tekniği seçimini etkileyen kriterlere gelen değerlerin veri tipleri incelenerek, sayısal karşılıkları belirlenmiştir. Bu işlemin amacı, veri setindeki değerlerin aynı tipte olmasını sağlayarak, makine öğrenmesi süreçlerinin etkinliğini arttırmaktır. Kriterlere atanan değerlerin belirlenmesinde ise uzman görüşlerine başvurularak tamamen rastsal değerler yerine daha tutarlı on bin satırlık bir veri seti oluşturulmuştur. Bu amaç için, uzmanlardan gelen 6 adet gerçek hayatta karşılaşabilecek önerme, veri setindeki değerlere uygulanarak veriler ve o verilere uygun işaretlenen yıkım kararları daha tutarlı hale getirilmiştir. Bu önermeler yöntemde detaylı açıklanmakla birlikte

patlayıcı ile yıkım tekniği kullanılamayacak durumları, yıkım makinesi kullanımı gerektiren durumları, kat eksiltme kararı gerektiren durumları, vinç kurulum alanı olmayan durumları, el ile kat eksiltme gerektiren durumları, araç trafiğinin mevcut olduğu ve farklı çevre parametrelerinin ele alındığı durumları örneklemektedir ve literatürle uyumlu gözükmektedir.

Veri seti oluşturulmasının ardından, elde edilen sonuçlar üzerinden analizler gerçekleştirilmiştir. Harrington' un 2012 yılında çalışmaya göre, sınıflama, belirli özelliklerdeki bir örneğin, hangi sınıfta olduğunun tahmin edilmesidir. Bu bilgiye dayanarak yapılan analizler sonucunda, elde edilen veri setindeki verilerle birlikte yıkım tekniği kararı verebilmenin bir sınıflandırma problemi olduğu sonucuna varılmıştır. Köksal'ın 2020 yılında yaptığı çalışmada, denetimli makine öğrenmesi yöntemi, etiketlenmiş veriler aracılığı ile eğitim gerçekleştirilir. Etiketlenmiş veri, kriterlerin değerlerine göre, aslında bilinen sonucun etiketlenmesi anlamına gelmektedir denilmektedir. Bu duruma yıkım teknikleri açısından bakıldığında, her teknik bir sınıf olarak değerlendirilmiş ve veri setindeki veriler literatür araştırmasından elde edilen yıkım tekniklerinin, belirlenen sayısal karşılıkları ile etiketlenmiştir.

Pedregosa ve arkadaşlarının belirlediği makine öğrenmesi modelleri seçimi yol haritasına göre, veri setindeki verilerin nitelikleri ve sayıları dikkate alınarak;

- Doğrusal destek vektör makineleri,
- K en yakın komşu algoritması,
- Destek vektör sınıflandırıcısı,
- Olasılıksal dereceli azalma yöntemi,
- Çekirdek yaklaşımı yöntemi

Sınıflandırma algoritmaları ve yöntemleri ile eğitimler gerçekleştirilerek sonuçları incelenmiştir.

Eğitimler sonucunda elde edilen modeller kullanılarak veri setindeki gerçek değerler ve modelin tahmin ettiği değerler karşılaştırılmıştır. Makine öğrenmesi süreçlerinde özelliklerin ölçeklendirilmesi önemli bir aşamadır.

Bollegala'nın 2017 yılında yaptığı çalışmaya göre, farklı özelliklere gelen değerler, değişken aralıklarda olabileceğinden, etkili bir öğrenme için özelliklerin ölçeklendirilerek aynı aralıklara dahil edilmesi gerekmektedir. Bu noktadan hareketle, her bir algoritma ve yöntem için işletilen tahmin ve gerçek değerlerin karşılaştırılması sürecinde, veri ölçeklenmesi ile yapılan eğitimlerin daha yüksek tahmin doğruluk skorları elde ettiği görülmüştür. Literatürden beslenerek elde edilen bu bulgu, makine öğrenmesi süreçlerinde veri ölçeklendirmenin önemini ortaya koymaktadır.

Feurer, Springenberg, ve Hutter'ın 2015 yılında yaptıkları çalışmaya göre, makine öğrenmesi süreçlerinde, öğrenme etkinliğinin arttırılabilmesi için, hiperparametre optimizasyonu yapılması gerekmektedir. Farklı hiperparametre değerleri ile farklı sonuçlar elde edebilmek mümkündür. Tahmin doğruluğunu arttırabilmek için, veri ölçeklendirme işlemi ile birlikte, her makine öğrenmesi algoritması ve yönteminin eğitim için kullandığı hesaplama yöntemlerini seçmekte kullanılan hiperparametrelerin de optimizasyonları yapılmıştır. Optimizasyon yapılamadan önceki varsayılan parametre değerleri ve optimizasyon sonrası seçilen parametre değerleri ile yapılan eğitimler sonucu, tahmin doğruluk oranları karşılaştırılarak en iyi skorlar aranmıştır.

Üretilen on bin satırlık veri seti ile yapılan eğitimlerde en iyi tahmin skorunun k en yakın komşu algoritması ile alındığı tespit edilmiştir. Pedregosa ve arkadaşlarının önerdiği makine öğrenmesi model seçimi yol haritasına göre, yüz bin veri üstü için eğitim yöntemleri olarak olasılıksal dereceli azalma yöntemi ve çekirdek yaklaşımı yöntemi önerilmiştir. Bu öneri doğrultusunda, veri seti üretimi süreçleri tekrar işletilerek, yüz bin on satırlık bir veri seti elde edilmiştir. Bu veri setinin oluşturulmasında da önceki süreçlerde kullanılan 6 adet önerme aynı Şekilde kullanılmıştır. Önerilen yöntemlerle yapılan eğitimler ve tahmin doğruluğu skorları ölçümlerinde, bu yöntemlerin başarımlarının, diğer algoritmalara göre düşük olduğu sonucuna varılmıştır.

Yapılan eğitim ve tahmin doğruluğu skorları ölçüm işlemleri sonucunda, k en yakın komşu algoritmasından elde edilen 0.8971'lik skor, en yüksek tahmin doğruluk skoru olarak belirlenmiştir. Buna göre, uzmanlar tarafından girilecek veriler için tahmin yapabilmek adına, k en yakın komşu algoritması ile yapılan eğitim sonucu elde edilen model kaydedilerek kullanıma hazır hale getirilmiştir.

Eğitilmiş modelin, uzmanlar tarafından kullanılabilmesi adına, web tabanlı bir yazılım geliştirilmiştir. Bu yazılımın web tabanlı olmasının nedeni, platform bağımsız olarak erişilebilir olmasıdır. Yazılım ara yüzü tüm platformlarda sorunsuz görüntülenebilecek Şekilde güncel kodlama teknikleri ile geliştirilmiştir. Bu yazılım temelde iki kullanım senaryosu üstüne kurgulanmıştır. Bu senaryolardan ilki, yıkım uzmanlarının yıkılacak binanın verilerini girerek, eğitilmiş model üstünden yıkım kararı desteği aldıkları senaryo, diğeri ise yıkım uzmanlarının veri tabanına yıkılacak olan binanın verileri ile birlikte, yıkım tekniği önerisi de girdikleri senaryo olarak tanımlanabilir. İkinci senaryodaki amaç, uzmanların veri tabanına katkı yapmalarını sağlamaktır ancak bu noktada girilen verilerin ve yıkım kararı önerilerinin, ülkemizde uygulanan regülasyonlara uygun olması gerekmektedir. Uzmanların yaptıkları katkıların artması ile birlikte veri sayısı artan veri seti kullanılarak yeniden makine öğrenmesi süreçleri işletilerek model oluşturulacak ve iteratif Şekilde ilerleyen bu süreç ile birlikte tahmin doğruluklarının da artacağı düşünülmektedir. Bununla birlikte, bundan sonraki çalışmalarda bir mobil uygulama geliştirilerek, veri girişleri noktasında, gps, lazer ölçüm cihazları, harita yazılımları gibi teknoloji destekli sistemlerden faydalanılmasının etkili olacağı söylenebilir.

Bütün çıktılar ışığında ele alındığında makine öğrenmesinin bina yıkım kararı süreçlerinde bir destek sistemi olarak kullanılabileceği sonucuna varılmıştır. Çok kriterli karar verme problemlerinde makine öğrenmesi model ve yöntemlerinin kullanımı ile geçmiş verilerden elde edilen tecrübelerle öğrenen sistemlerle birlikte karar desteği sağlanabilmektedir. Bu tür çalışmalarda, eğitim için kullanılacak veri setinin tutarlı olması ile etkin kararlar verileceği, verilerin ölçeklendirilmesi ile birlikte, kullanılan parametre optimizasyon tekniklerinin de eğitim başarımını arttırdığı sonucuna ulaşılmıştır. Bu süreç içerisinde en önemli girdinin tutarlı bir veri seti olduğu söylenebilir. Gerçek verilere dayalı setlerin olmaması durumunda üretilecek sentetik veri setlerinde uzman görüşlerinin önemli olduğu sonucuna varılmıştır. Bu sayede tutarlı veriler üretilerek daha efektif öğrenme sonuçları elde edilmiştir. Makine öğrenmesi süreçlerinde, modellerin kullanımı ve optimizasyonları için python programlama dili ile geliştirilen kütüphaneler kullanılmıştır. Bu kapsamda, girdi olarak kullanılan verilerin niteliklerine uygun makine öğrenmesi modeli seçiminin önemli olduğu sonucuna ulaşılmıştır.

Bu alanda veri seti oluşturulmasıyla, bundan sonraki çalışmalar için de yol gösterici olacağı düşünülmektedir. Bu çerçeveden bakıldığında oluşturulan veri setinin açık platformlarda

paylaşılması ve ulaşılabilir hale getirilmesinin araştırmacılara ve bu alanda çalışmak isteyenlere katkı sağlayacağı düşünülmektedir. Bundan sonra bu alanda oluşturulacak veri setlerine yıkım maliyeti, yıkım firmasının yıkım makineleri envanteri, adres kodu gibi kriterler de eklenerek veri setleri geliştirilebilir. Bu geliştirmeler ile yıkım kararı verilirken eklenen kriterlerin de etkileri sonuca yansıtılarak karar desteğinde iyileştirmelerin yapılabileceği düşünülmektedir.

Yıkım tekniklerinin seçiminde kullanılmak üzere geliştirilen veri setine ulaşma noktasında diğer bir öneri ise, ülkemizde uygulanan başarılı yıkım projelerindeki verilerin toplanarak, uygun yıkım kararları ile işaretlendikten sonra araştırmacıların kullanıma açılmasıdır. Bu anlamda ülkemizde son yıllarda gündemde olan kentsel dönüşüm projelerinde uygulanan teknikler toplanarak veri seti oluşturma süreçlerine başlanması, bu alanda eksik olan gerçek verilerin de ulaşılabilir hale getirilerek, gerekirse akademik verilerden de beslenerek ortaya bir veri setinin çıkması sağlanabilir.

Veri setlerinin geliştirilmesi ve çok miktarda verinin toplanması ile birlikte, veriler arasındaki ilişkilerin daha efektif bir Şekilde ortaya çıkarılarak, daha iyi karar desteği konusunda derin öğrenme yöntemleri de kullanılabilir.

Yıkım süreçlerinde önemli olan noktalardan bir tanesi de seçici yıkımdır. Seçici yıkım ile yıkılacak yapıdan elde edilen kullanılabilir malzemelerin geri dönüşümleri sağlanabilmektedir. Bu çerçeveden bakıldığında, bundan sonra yapılacak çalışmalarda yıkılacak olan yapının iç ve dış görüntülerinin sisteme yüklenerek, görüntü işleme teknikleri ile geri dönüştürülecek malzemelerin analizi yapılarak yıkım uzmanlarına raporlanmasının mümkün olacağı düşünülmektedir.

KAYNAKLAR

Abanuz, F. (2005). Eskimiş Betornarme Yapılarda Yıkımın Planlanması, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 1-1.

Agarwal, P., Sahai, M., Mishra, V., Bag, M., and Singh, V. (2014). Supplier Selection in Dynamic Environment using Analytic Hierarchy Process. I.J. Information Engineering and Electronic Business,4,20-26.

Akman, E., Karaman, A. S., ve Kuzey, C. (2020). Visa trial of international trade: evidence from support vector machines and neural networks. Journal of Management Analytics, 7(2), 231–252.

Anumba, C., Abdullah, A., and Fesseha, T. (2003). Selection of demolition techniques: a case study of the Warren Farm Bridge. Structural Survey, 21(1), 36-48.

Anumba, C.J., Abdullah, A., and Ruikar, K. (2008) An integrated system for demolition techniques selection. Architectural Engineering and Design Management, 4(2), 130-148.

Arham, A. (2003). Intelligent Selection of Demolition Techniques, Doktora Tezi, Loughborough University, Loughborough,1-119.

Ayhan, S., ve Erdoğmuş, Ş. (2014). Destek Vektör Makineleriyle Sınıflandırma Problemlerinin Çözümü İçin Çekirdek Fonksiyonu Seçimi. Eskişehir Osmangazi Üniversitesi İibf Dergisi, 9(1), 175-198.

Băzăvan, E. G., Li, F., and Sminchisescu, C. (2012). Fourier kernel learning. European Conference on Computer Vision, 459-473.

Bhatnagar, A., ve Srivastava, S. (2019). A Robust Model for Churn Prediction using Supervised Machine Learning, 2019 IEEE 9th International Conference on Advanced Computing (IACC), 45-49.

Brickley, D., Burgess M., and Noy N. (2019). Google Dataset Search: Building a search engine for datasets in an open Web ecosystem. World Wide Web Conference (WWW '19). Association for Computing Machinery, New York, NY, USA, 1365–1375.

Burges, C.J.C. (1998). A Tutorial On Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 2(2), 121-167.

Cables E., Lamata, M.T., and Verdegay, J.L. (2016). RIM-reference ideal method in multicriteria decision making. Inf. Sciences, 337-338, 1-10.

Carlson, C., ve Fuller, R. (1996), Fuzzy multiple criteria decision making: recent developments, Fuzzy Sets and Systems, 78(2), 139-153.

Chen, Z., Abdullah, A. B., Anumba, C. J., and Li, H. (2014). ANP experiment for demolition plan evaluation. Journal of Construction Engineering and Management, 140(2), 51-60.

Çetinyokuş, T., ve Gökçen, H. (2002). Borsada Göstergelerle Teknik Analiz İçin Bir Karar Destek Sistemi. Journal of the Faculty of Engineering & Architecture of Gazi University,17(1), 43-58.

Çevre ve Şehircilik Bakanlığı. (2004). Hafriyat Toprağı, İnşaat ve Yıkıntı Atıklarının Kontrolü Yönetmeliği. Resmi Gazete, 25406.

Gottschlich, J., Solar-Lezama, A., Tatbul, N., Carbin, M., Rinard, M., Barzilay, R., Amarasinghe, S., Tenenbaum, J.B. and Mattson, T. (2018). The three pillars of machine programming. Proceedings of the 2nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, 69-80.

Cortes, C., Mohri, M., and Talwalkar, A. (2010). On the Impact of Kernel Approximation on Learning Accuracy. Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS), 9, 113-120.

Dağdeviren M., Eraslan, E., Kurt, M., ve Dizdar, E.N. (2005). Tedarikçi seçimi problemine analitik ağ süreci ile alternatif bir yaklaşım, Teknoloji Dergisi, 8(2), 115-122.

Darmawan, I., Rahmatulloh, A., Nuralam, I. M. F., Rianto, and Gunawan, R. (2020). Optimizing Data Storage in Handling Dynamic Input Fields with JSON String Compression. 8th International Conference on Information and Communication Technology (ICoICT), Yogyakarta, Indonesia, 1-5.

Diven, R. J., and Shaurette, M. (2010). Demolition : Practices, Technology, and Management. Indiana: Purdue University Press, 1, 13.

Dixit R., Chinnam, R.B., and Singh H. (2020). Artificial Intelligence and Machine Learning in Sparse/Inaccurate Data Situations, 2020 IEEE Aerospace Conference, 1-8.

Druzdzel M.J., and Flynn R.R. (2011). Decision Support Systems. In Bates, M.J. (Ed.), Understanding Information Retrieval Systems : Management, Types, and Standards. Florida: CRC Press, 461-471.

Elias Özkan, S.T. (2003). Deconstruction of earthquake damaged buildings in Turkey, Deconstruction and Materials Reuse – Proceedings of 11th Rinker International Conference on Deconstruction and Materials Reuse, 287, 138-150.

Elias Özkan, S.T. (2012). Selective demolition of redundant and earthquake damaged buildings in Turkey. Orta Doğu Teknik Üniversitesi Mimarlık Fakültesi Dergisi, 2012(13620), 139–152.

Forcier, J., Bissex, P., and Chun, W. J. (2008). Python web development with Django. Boston:Addison-Wesley Professional, 6.

Feurer, M., Springenberg, J., and Hutter, F. (2015). Initializing bayesian hyperparameter optimization via meta-learning. In Proceedings of the AAAI Conference on Artificial Intelligence, 29(1), 1128.

Forman, E.H., and Selly, M.A. (2001). Decision by objectives: How to convince others that you are right. Singapore: World Scientific Co. Pte. Ltd., 1.

Frozza, A.A., Jacinto, S. R., and Mello, R. d. S. (2020). An Approach for Schema Extraction of NoSQL Graph Databases. IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI), Las Vegas, NV, USA, 271-278.

Genç, F. N. (2008). Türkiye'de kentsel dönüşüm: Mevzuat ve uygulamaların genel görünümü. Yönetim ve Ekonomi: Celal Bayar Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 15(1), 115-130.

Geneves, P., Layaida, N. and Quint, V. (2012). On the analysis of cascading style sheets. In Proceedings of the 21st international conference on World Wide Web, 809-818.

Ghosh S., Dasgupta, A., and Swetapadma, A. (2019). A Study on Support Vector Machine based Linear and Non-Linear Pattern Classification, 2019 International Conference on Intelligent Sustainable Systems (ICISS), 24-28.

Goodman, D. (2007). JavaScript bible (Fourth edition). Indianapolis: Wiley Publishing,Inc., 17.

Görgülü, Z. (2009). Kentsel dönüşüm ve ülkemiz. TMMOB İzmir Kent Sempozyumu, 769-780.

Harrington, P. (2012). Machine Learning in Action. New York: Manning Publications, 8-9.

Hickson, I., and Hyatt, D. (2011). HTML 5. W3C Working Draft WD-html5-20110525, 53.

Ionescu, C., Popa, A.I., and Sminchisescu, C. (2017). Large-Scale Data-Dependent Kernel Approximation. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), 54, 19-27.

İnternet: Sarkar, T. (2018). Synthetic data generation — a must-have skill for new data scientists. URL: https://towardsdatascience.com/synthetic-data-generation-a-must-have-skill-for-new-data-scientists-915896c0c1ae, Son Erişim Tarihi: 02.07.2020.

İnternet: 5366 Sayılı Yıpranan Tarihi ve Kültürel Taşınmaz Varlıkların Yenilenerek Korunması ve Yaşatılarak Kullanılması Hakkında Kanun. URL: https://www.mevzuat.gov.tr/mevzuat?MevzuatNo=5366&MevzuatTur=1&MevzuatTertip=5, Son Erişim Tarihi: 01.01.2021.

İnternet: 5393 Sayılı Belediye Kanunu. URL: https://www.mevzuat.gov.tr/mevzuat?MevzuatNo=5393&MevzuatTur=1&MevzuatTertip=5, Son Erişim Tarihi: 01.01.2021.

İnternet: 6306 Sayılı Afet Riski Altındaki Alanların Dönüştürülmesi Hakkında Kanun. URL: https://www.mevzuat.gov.tr/mevzuat?MevzuatNo=6306&MevzuatTur=1&MevzuatTertip=5, Son Erişim Tarihi: 01.01.2021.

İnternet: The Balance Small Bussines (2019). URL: https://www.thebalancesmb.com/ways-to-demolish-buildings-844420, Son Erişim Tarihi: 07.02.2021.

Jayashree, G., and Priya, C. (2020). Data Integration with XML ETL Processing, 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA), Gunupur, India, 1-8.

Kavzoğlu, T. ve Çölkesen, İ. (2010). Destek vektör makineleri ile uydu görüntülerinin sınıflandırılmasında kernel fonksiyonlarının etkilerinin incelenmesi. Harita Dergisi, 144(7), 73-82.

Keramati, A., Jafari-Marandi, R., Aliannejadi, M., Ahmadian, I., Mozaffari, M., and Abbasi, U. (2014). Improved churn prediction in telecommunication industry using data mining techniques, Applied Soft Computing, 24, 994-1012.

Koca, O. (2006). Patlayıcı Maddelerle Kontrollü Yapı Yıkımı, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 4-6.

Koçak, H., Tolanlar, M. (2008). Kentsel Dönüşüm Uygulamaları (Aydın Ve Afyonkarahisar Örnekleri). Afyon Kocatepe Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 10(2), 397-415.

Köksal, Ö. (2020). Tuning the Turkish Text Classification Process Using Supervised Machine Learning-based Algorithms, 2020 International Conference on INnovations in Intelligent SysTems and Applications (INISTA), 1-7.

Kreibich, J. (2010). Using SQLite. California: O'Reilly Media, Inc., 1-2.

Lutz, M. (2001). Programming python (Second edition). California: O'Reilly Media, Inc., 1.

Mahmud, S. M. H., Hossin, M. A., Jahan, H., Noori, S. R. H. and Bhuiyan, T. (2018). CSV-ANNOTATE: Generate annotated tables from CSV file, 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, 71-75.

Michalski, R.S., Carbonell, J.G., and Mitchell, T.M. (1983). Machine Learning An Artificial Intelligence Approach. Kaliforniya: Tioga Publishing Co., 25-28.

MurtiRawat, R., Panchal, S., Singh, V.K. and Panchal, Y. (2020). Breast Cancer Detection Using K-Nearest Neighbors, Logistic Regression and Ensemble Learning, 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), 534-540.

Orman, A., Düzkaya, H., Hayri, U. L. V. İ., & Akdemir, F. (2018). Multi-criteria evaluation by means of using the analytic hierarchy process in transportation master plans: Scenario selection in the transportation master plan of Ankara. Gazi University Journal of Science, 31(2), 381-397.

Oladipupo Ayodele, T. (2010). Types of Machine Learning Algorithms. New Advances in Machine Learning,Zhang Y. (Editor), Rijeka:Intech, 19-48.

Öcal, C, İnce, H. (2012). Türkiye'de Mevcut Yapı Stoğu Ve Kentsel Dönüşüm. Uluslararası Teknolojik Bilimler Dergisi, 4(2), 89-95.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P.,Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python, Journal of Machine Learning,12, 2825-2830.

Pourkamali-Anaraki, F., Becker, S., and Wakin, M. (2018). Randomized clustered nystrom for large-scale kernel machines. Proceedings of the AAAI Conference on Artificial Intelligence, 32(1), 3960-3967.

Rahman A., and Sungyoung L., Tae C.C. (2017). Accurate multi-criteria decision making methodology for recommending machine learning algorithm. Expert Systems with Applications,71,257-278.

Rohland, L. (2017). Analytic hierarchy process (AHP). Salem Press Encyclopedia.

Sadioğlu, U., Tiryaki, V., ve Korkmaz, A. (2016). Altındağ Belediyesi örneği üzerinden Türkiye'de kentsel dönüşüm politikasının değerlendirilmesi. Ankara Üniversitesi SBF Dergisi, 71(3), 757-796.

Saito S., Yuyama, K., Ishii, M., Huang, V., and Nishi, H. (2020). Representation of Plant Structure using XML and Its Application to Cultivation Management, 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vienna, Austria, 125-131.

Shalev-Shwartz S., and Ben-David S. (2014). Understanding Machine Learning: From Theory to Algorithms. New York: Cambridge University Press, 21-22.

Sipahi, S., Timor, M. (2010). The analytic hierarchy process and analytic network process: an overview of applications. Management Decision, 48(5), 775-808.

Soman, K. P., Loganathan, R., and Ajay, V. (2009). Machine learning with SVM and other kernel methods. New Delhi: PHI Learning Pvt. Ltd., 1.

Şahmaran, M., Özgür, A., Gürbüz, A., ve Koçkar, M. K. (2015). Yıkım Tekniklerinin Araştırılması ve Uygulamaları El Kitabı. Ankara: Gazi Üniversitesi, 1,9,28,34.

Tavsan V., Türker T. (2020). Controlled demolition techniques and demolition direction. Sciennovation A Journal of Structural Science and Innovation, 1(2), 24-25.

Toğay, A. (2002). Ahşap Yapılar, Türkiye'de Ahşap Yapı Endüstrisinin Durumu, Sorunları ve Çözüm Önerileri, Doktora Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara,1.

Turban E., Jay E. and Ting-Peng L. (2007). Decision support sytems and intelligent systems (Seventh Edition). New Delhi: Asoke K. Ghosh, Prentice-Hall of India Private Limited,10-11.

Uzun E. (2020). A Novel Web Scraping Approach Using the Additional Information Obtained From Web Pages, in IEEE Access, 8,61726-61740.

Vaughan-Nichols, S. J. (2010). Will HTML 5 Restandardize the Web?, in Computer, 43(4), 13-15.

Xiao F. (2020). EFMCDM: Evidential Fuzzy Multicriteria Decision Making Based on Belief Entropy, IEEE Transactions on Fuzzy Systems, 28(7), 1477-1491.

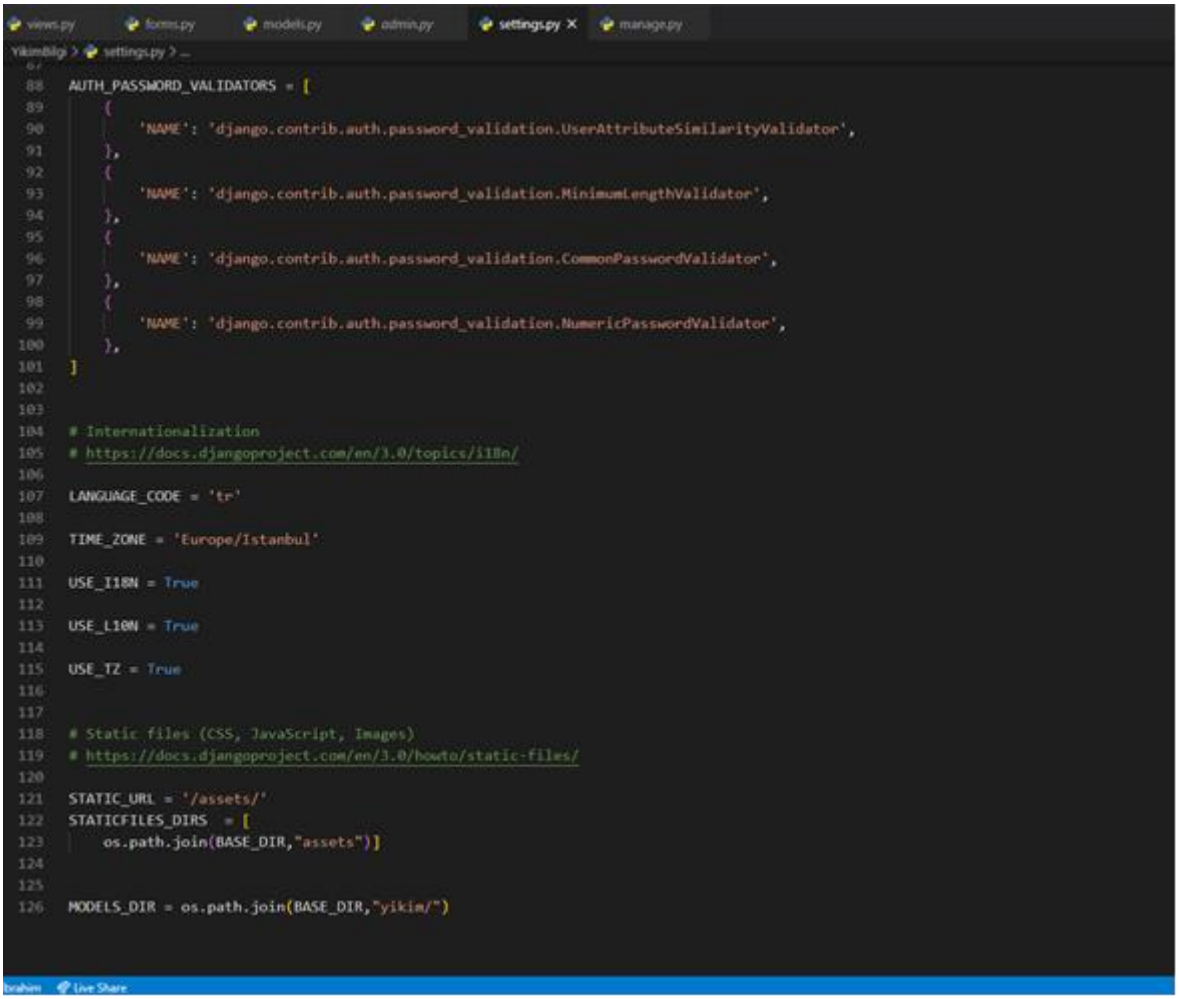
Yang, L., Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. Neurocomputing, 415, 295.

Yılmaz, Z., Çankaya, F., Karakaya, A. (2017). Yıkım ve yeniden yapım maliyetlerini etkileyen faktörlerin bina maliyet oranı açısından önemi. ODÜ Sosyal Bilimler Araştırmaları Dergisi (ODÜSOBİAD),7(2),395-412.

Zelle, J. M. (2004). Python programming: an introduction to computer science. Oregon:Franklin, Beedle & Associates, Inc., 9.

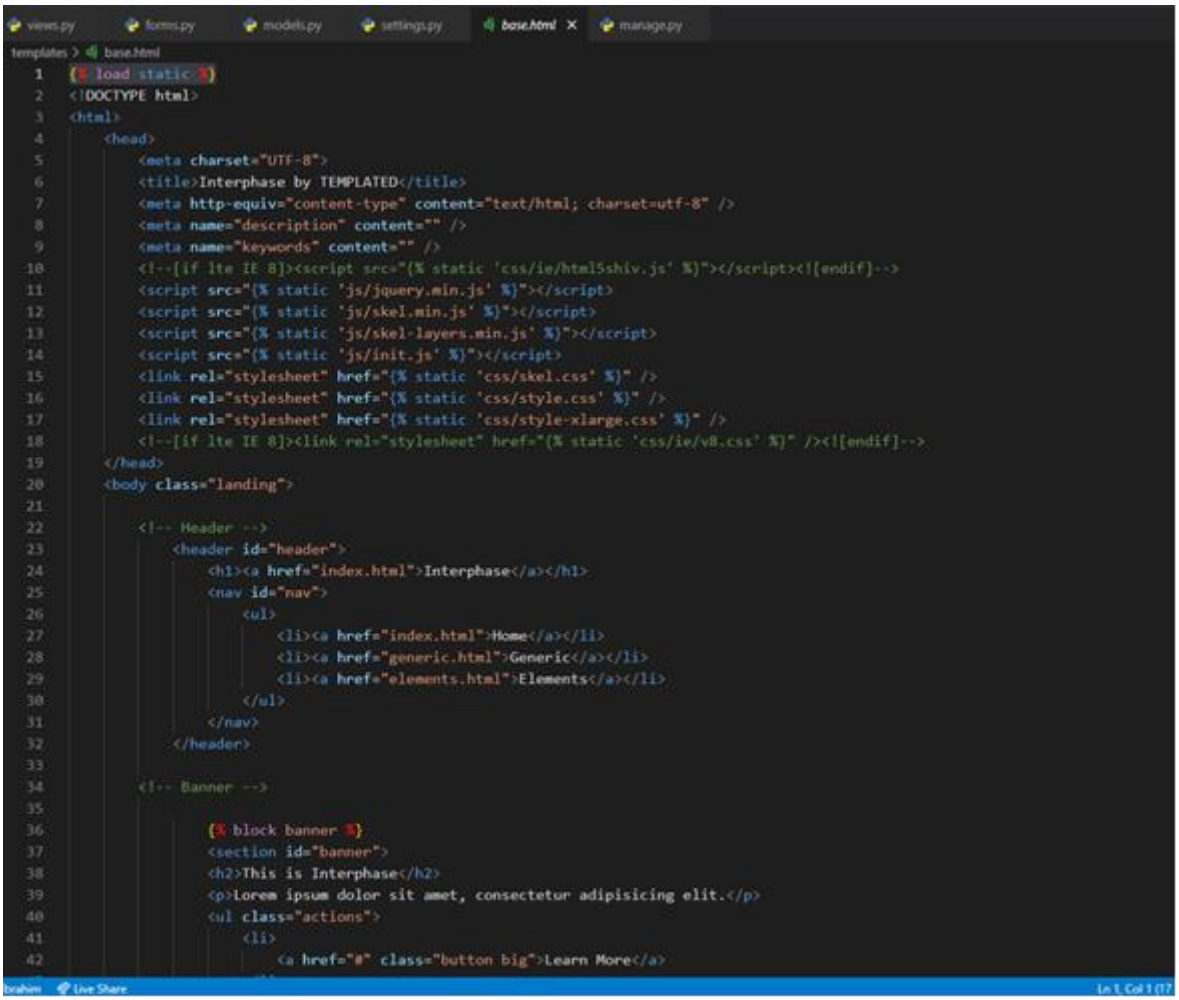
EKLER

EK-1. Django ayarları



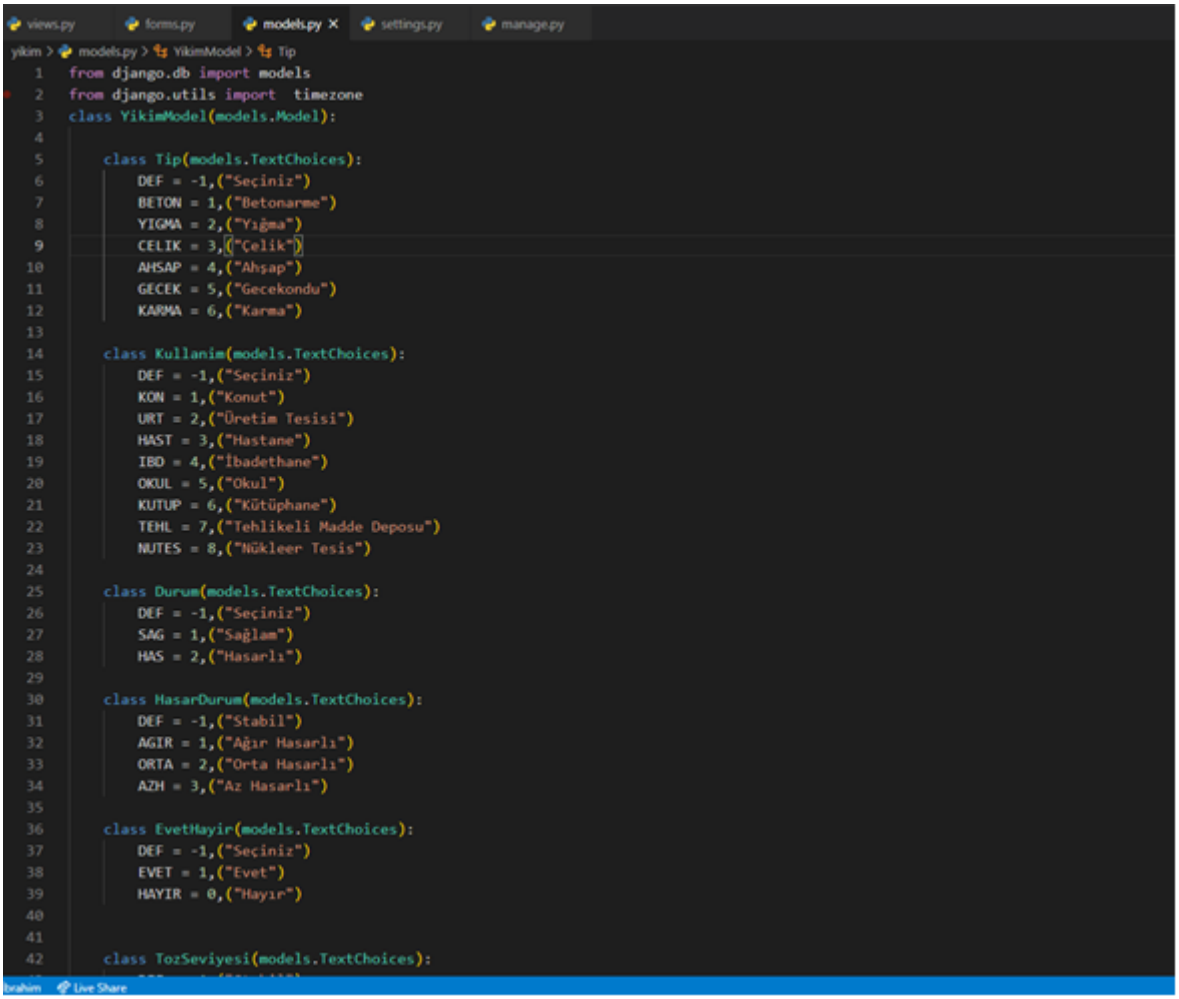
Şekil 1.1. Django ayarları kodları ekran görüntüsü

EK-2. Web uygulaması ara yüzü html kodları



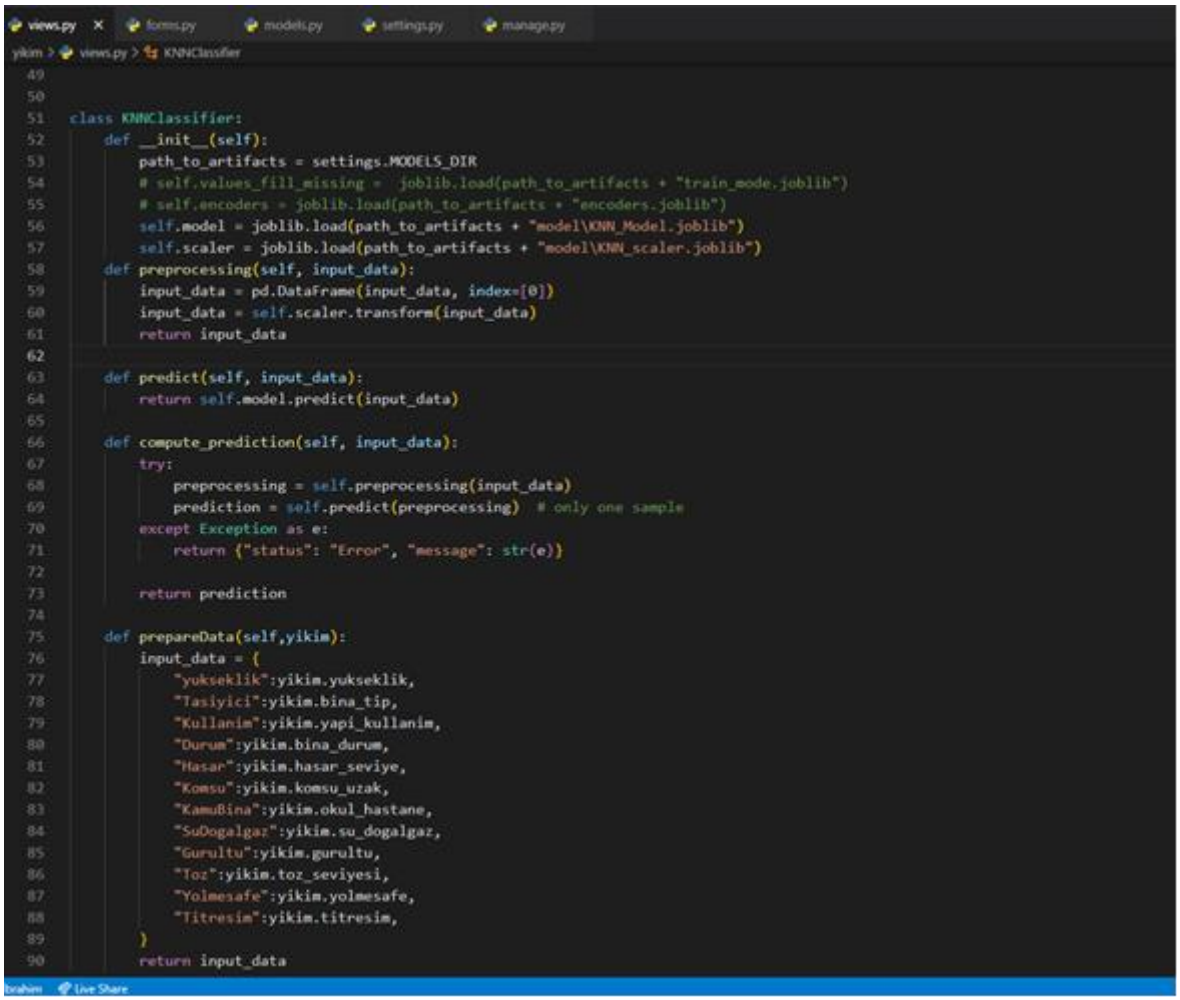
Şekil 2.1. Web uygulaması ara yüzü html kodları ekran görüntüsü

EK-3. Yıkılacak yapı özellikleri sınıfları kodları



Şekil 3.1. Yıkılacak yapı özellikleri sınıfları kodları ekran görüntüsü

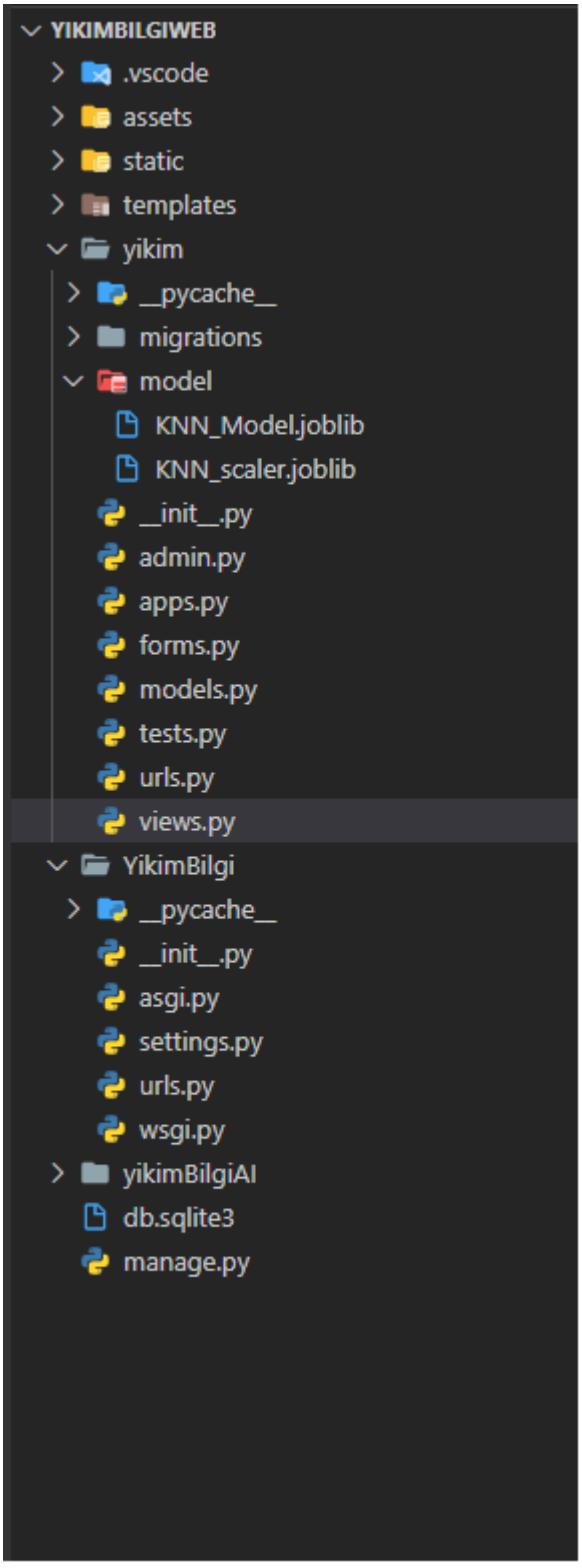
EK-4. K en yakın komşu algoritması ile tahmin sınıfı



```python
class KNNClassifier:
    def __init__(self):
        path_to_artifacts = settings.MODELS_DIR
        # self.values_fill_missing =  joblib.load(path_to_artifacts + "train_mode.joblib")
        # self.encoders = joblib.load(path_to_artifacts + "encoders.joblib")
        self.model = joblib.load(path_to_artifacts + "model\KNN_Model.joblib")
        self.scaler = joblib.load(path_to_artifacts + "model\KNN_scaler.joblib")
    def preprocessing(self, input_data):
        input_data = pd.DataFrame(input_data, index=[0])
        input_data = self.scaler.transform(input_data)
        return input_data

    def predict(self, input_data):
        return self.model.predict(input_data)

    def compute_prediction(self, input_data):
        try:
            preprocessing = self.preprocessing(input_data)
            prediction = self.predict(preprocessing)  # only one sample
        except Exception as e:
            return {"status": "Error", "message": str(e)}

        return prediction

    def prepareData(self,yikim):
        input_data = {
            "yukseklik":yikim.yukseklik,
            "Tasiyici":yikim.bina_tip,
            "Kullanim":yikim.yapi_kullanim,
            "Durum":yikim.bina_durum,
            "Hasar":yikim.hasar_seviye,
            "Komsu":yikim.komsu_uzak,
            "KamuBina":yikim.okul_hastane,
            "SuDogalgaz":yikim.su_dogalgaz,
            "Gurultu":yikim.gurultu,
            "Toz":yikim.toz_seviyesi,
            "Yolmesafe":yikim.yolmesafe,
            "Titresim":yikim.titresim,
        }
        return input_data
```

Şekil 4.1. K en yakın komşu algoritması ile tahmin sınıfı ekran görüntüsü

EK-5. Web tabanlı uygulama klasör yapısı



Şekil 5.1. Web tabanlı uygulama klasör yapısı ekran görüntüsü

EK-6. Yıkım uzmanları için yarı yapılandırılmış görüşme formu

Anketi cevaplayana ilişkin sorular	
Adınız Soyadınız	
Göreviniz	
Sektördeki Deneyim Süreniz	0-5 yıl: [ ] 5-10 yıl: [ ] 10-20 yıl: [ ] Diğer: [ ]
Yaşınız	
İletişim Numaranız	
Kurum Bilgileri	
Kurum Adı	
Kurum Türü	Kamu Kurumu: [ ] Özel Sektör: [ ] Üniversite: [ ]
Yıkım süreçleri ile ilgili sorular	
1. Yıkım teknikleri kararı verirken hangi değişkenlerden faydalanmaktasınız?	
2. Yıkım teknikleri seçiminde en önemli kriter sizce hangisidir?	Yapı Yüksekliği: [ ] Çevresel Faktörler: [ ] Taşıyıcı Sistem Türü: [ ] Diğer: [ ]
2. soruya cevabınız Diğer ise açıklamasını yapınız.	
4. Yıkım tekniği seçiminde kullandığınız yazılım ya da donanım teknolojileri nelerdir?	
5. Önceki başarılı ya da başarısız yıkımlardan elde edilen bina özellikleri ve bu özelliklere göre hangi yıkım tekniğinin kullanıldığına dair elinizde veriler var mıdır?	Evet: [ ] Hayır: [ ]

EK-6. (devam)Yıkım uzmanları için yarı yapılandırılmış görüşme formu

5. soruya cevabınız Evet ise topladığınız verilerden bina ve çevre kriterlerine göre verdiğiniz yıkım kararına ilişkin yıkım senaryosu örneği veriniz.	
6. Yıkım sırasında oluşabilecek riskler nelerdir?	
7. Yıkım sırasında oluşabilecek tehlikeli durumlar için teknik seçiminiz değişkenlik göstermekte midir? Örnek ile açıklayınız.	

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	:İbrahim Cihan YETİKEN
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 25.11.1982, Ankara
Medeni hali	:
Telefon	:
Faks	:
e-mail	:

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Doktora	Gazi Üniversitesi / Endüstriyel Teknoloji Eğt	Devam ediyor
Yüksek lisans	Gazi Üniversitesi / Endüstriyel Teknoloji Eğt	2010
Lisans	Gazi Üniversitesi / Endüstriyel Teknoloji Öğrt.	2006
Lise	Gazi Anadolu Teknik Listesi	2001

İş Deneyimi

Yıl	Yer	Görev
2011	Halen Gazi Üniversitesi	Öğretim Görevlisi
2008-2011	CG Yazılım	Yazılım Geliştirme Uzmanı

Yabancı Dil

İngilizce

Yayınlar

Toğay, A., Akdur, T. E., Yetişken, İ. C., Ve Bilici, A. (2013). Eğitim süreçlerinde sosyal ağların kullanımı: Bir MYO deneyimi. XIV. Akademik Bilişim Konferansı, 28-30.

Büyüktürkmen, M., Yetişken İ.C. (2013). Kahramanmaraş Geleneksel El Sanatlarının Internet Ortamında Tanıtılmasına İlişkin Web Uygulaması Örneği. Uluslararası Türk ve Dünya Kültüründe Kahramanmaraş Sempozyumu,1.

Toğay, A., Yetişken, İ.C. (2014). Multimedia ve video teknolojileri., S. Eryılmaz ve H. Çakır (Editörler). Eğitimciler İçin Bilişim Teknolojileri, Ankara. Pegem A Yayıncılık, 364-380.

Hobiler

Gitar, fotoğrafçılık

GAZİ GELECEKTİR..