**ANKARA ÜNİVERSİTESİ**

**FEN BİLİMLERİ ENSTİTÜSÜ**

**SEMİNER RAPORU**

**ASTRONOMİDE MAKİNE ÖĞRENMESİ UYGULAMALARI**

**Bedri KESKİN**

**ASTRONOMİ VE UZAY BİLİMLERİ ANABİLİM DALI**

**ANKARA**

**2021**

**Her hakkı saklıdır**

**İÇİNDEKİLER**

[SİMGELER DİZİNİ iii](#_Toc92350176)

[ŞEKİLLER DİZİNİ iv](#_Toc92350177)

[1. MAKİNE ÖĞRENMESİ 1](#_Toc92350178)

[**1.1. Denetimli Öğrenme (ing. Supervised Learning) 3**](#_Toc92350179)

[**1.2. Denetimsiz Öğrenme (ing. Unsupervised Learning) 5**](#_Toc92350180)

[**1.3. Pekiştirmeli Öğrenme (ing. Reinforcement Learning) 7**](#_Toc92350181)

[**1.4. Makine Öğrenmesi Süreci 8**](#_Toc92350182)

[**1.5. Yetersiz Öğrenme (ing. Underfitting) ve Aşırı Öğrenme (ing. Overfitting) 9**](#_Toc92350183)

[**1.6. Makine Öğrenmesi Kütüphaneleri 10**](#_Toc92350184)

[**1.7. Makine Öğrenmesi Veri Kaynakları 10**](#_Toc92350185)

[2. ASTRONOMİDE MAKİNE ÖĞRENMESİ UYGULAMALARINDAN ÖRNEKLER 11](#_Toc92350186)

[3. ÖRNEK MAKİNE ÖĞRENMESİ UYGULAMASI 13](#_Toc92350187)

[KAYNAKLAR 16](#_Toc92350188)

SİMGELER DİZİNİ

f X ile y arasındaki ilişki

X Makine öğrenmesinde girdi veri kümesi

y Makine öğrenmesinde çıktı veri kümesi

**Kısaltmalar**

GBDT Gradient Boosting Decision Trees

KCOI Kepler Cumulative Object of Interest

kNN k Nearest Neighbors

KOI Kepler Objects of Interest

LDA Linear Discriminant Analysis

NMF Non-negative Matrix Factorization

PCA Principal Component Analysis

SVM Support Vector Machines

TESS Transiting Exoplanet Survey Satellite

TCE Threshold Crossing Events

ŞEKİLLER DİZİNİ

[Şekil 1.1 Yapay Zekâ / Makine Öğrenmesi ilişkisi (Anonymous 2021a) 1](#_Toc92349816)

[Şekil 1.2 Turing testi (Anonymous 2021b) 2](#_Toc92349817)

[Şekil 1.3 Makine öğrenmesinin 3 temel yöntemi (Anonymous 2021c) 3](#_Toc92349818)

[Şekil 1.4 X ile y arasındaki ilişki bir doğru ile temsil edilebilir 4](#_Toc92349819)

[Şekil 1.5 Sınıflandırma 4](#_Toc92349820)

[Şekil 1.6 X girdi veri setindeki veriler arasındaki ilişkilere göre kümeleme yapma 5](#_Toc92349821)

[Şekil 1.7 Boyut indirgeme 6](#_Toc92349822)

[Şekil 1.8 Pekiştirmeli öğrenme (Anonymous 2021d) 7](#_Toc92349823)

[Şekil 1.9 Makine öğrenmesi süreci (Anonymous 2021e) 8](#_Toc92349824)

[Şekil 1.10 Yetersiz öğrenme ve aşırı öğrenme (Anonymous 2021f) 9](#_Toc92349825)

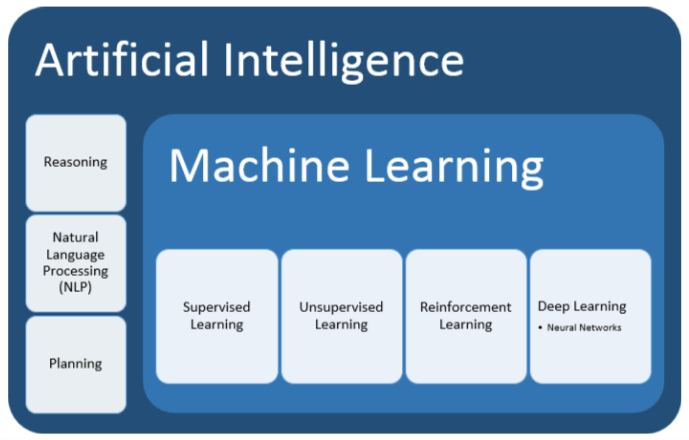
[Şekil 3.11 10 sınıfa ayrılmış galaksi imajlarından örnekler (Anonymous 2021m) 13](#_Toc92349826)

[Şekil 3.12 1000 çevrim ile elde edilen Doğruluk (ing. Accuracy) ve Kayıp (ing. Loss) grafikleri 14](#_Toc92349827)

[Şekil 3.13 Hata matrisi (ing. confusion matrix) 15](#_Toc92349828)

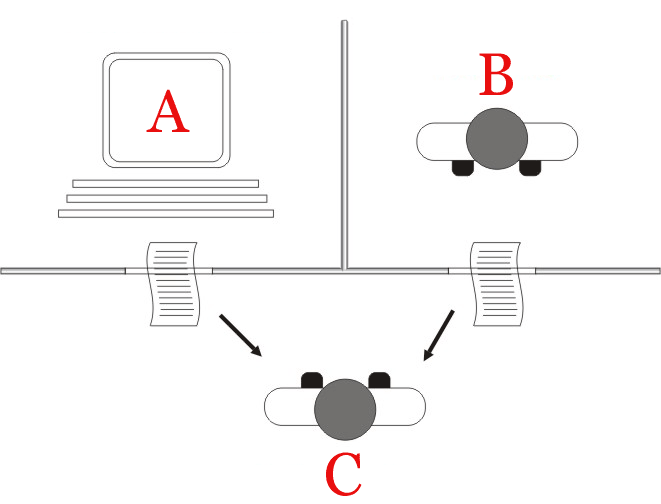
1. MAKİNE ÖĞRENMESİ

Makine öğrenmesi (ing. machine learning) yapay zekânın (ing. artificial intelligance) bir alt dalıdır (Şekil 1.1).



Şekil 1.1 Yapay Zekâ / Makine Öğrenmesi ilişkisi (Anonymous 2021a)

Makineler düşünebilir mi? Antik çağda Yunan, Çin vd. medeniyetlerde yapılan otomatik ardışık işlemler yapabilen otomatlar, 8. yy.da Musa kardeşlerin otomatik flüt çalgısı, 13. yy.da El-Cezeri’nin ilk insan benzeri mekanik cihazları, 15. yy.da Pascal’ın toplama-çıkarma ve tekrarlanan işlemlerle çarpma ve bölme yapabilen ilk hesap makinesi, Leibniz’in dört işlem yapabilen hesap makinesi düşünen makinelerin öncüleri olarak düşünülebilir. Yapay zekâ 1950’lerde ortaya çıkmıştır. Alan Turing’in (Turing 1959) “Computing Machinery and Intelligence” makalesi bir dönüm noktasıdır. Bu makalesinde Turing Testi olarak bilinen bir yapay zekânın başarılı olmasının ölçütünü ortaya koymuştur. Bir yapay zekâ sistemine ve bir insana sorular sorulur (Şekil 1.2). Verilen cevaplardan hangisinin yapay zekâ hangisinin insan olduğu ayırt edilemezse o yapay zekâ sistemi başarılıdır.



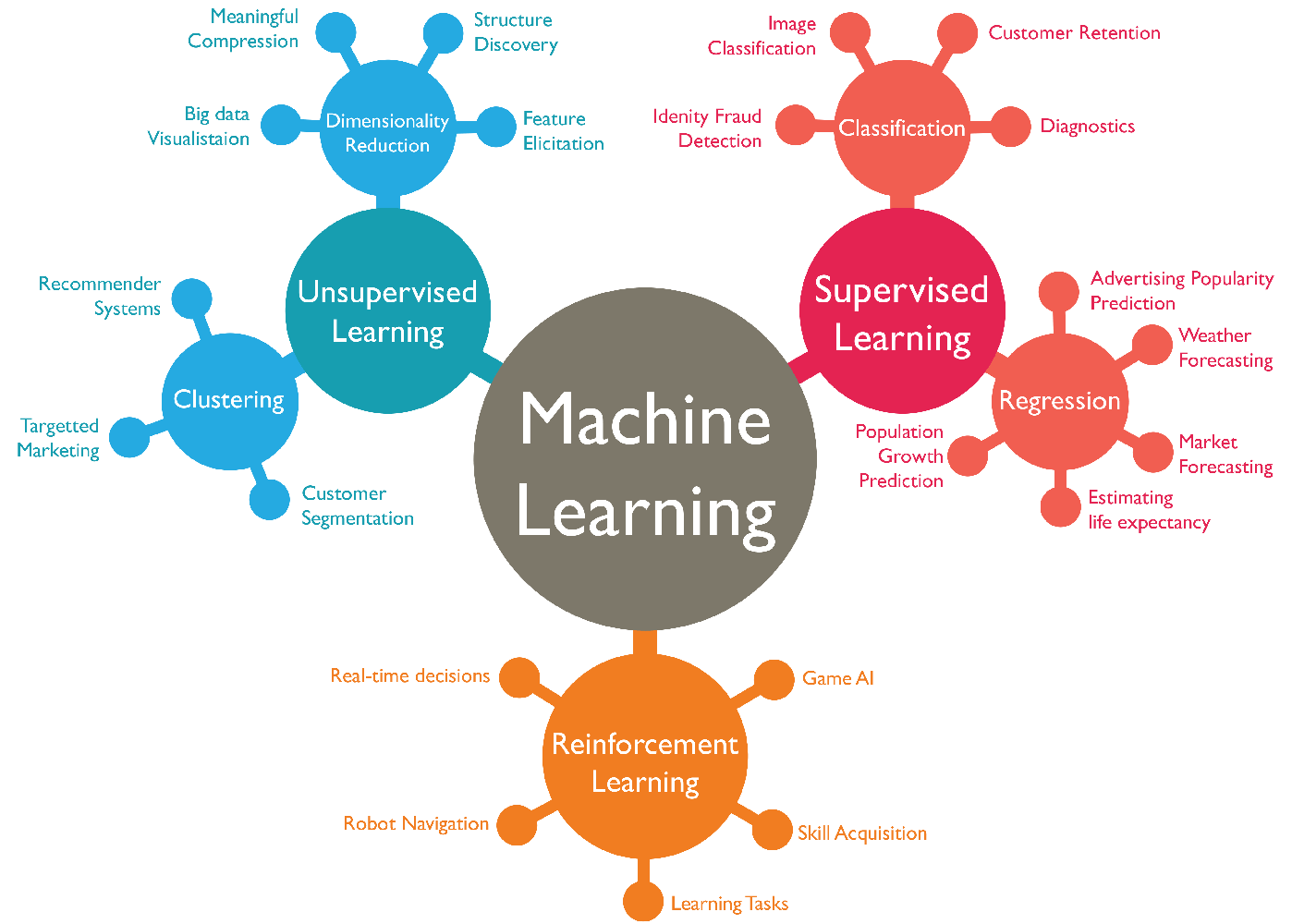
Şekil 1.2 Turing testi (Anonymous 2021b)

“Makine öğrenmesi” 1959’da IBM çalışanı Arthur Samuel önerisiyle literatüre girdi. Samuel, Lisp dilinde dama oynayan makine öğrenmesi uygulaması yaptı. 1960’ların sonunda yapay zekânın yüksek işlem gücü gerektirdiği fark edildi. O yıllarda bilgisayar teknolojisi yetersiz kalmaktaydı. Bu yüzden yapay zekâ rafa kalktı. Teknolojinin hızlı gelişmesiyle yapay zekâ çalışmaları 1980’lerde tekrar ivme kazandı. Günümüzde yapay zekâ/makine öğrenmesi resim/ses/yazı tanıma, robotlar, oyunlar, hava durumu tahmini, piyasa analizi gibi birçok alanda uygulanmıştır.

Makine öğrenmesinde temel prensip aşağıdaki şekilde ifade edilebilir:

* X girdi kümesi ile y çıktı kümesi arasında bağ kurma f:X→y
* Mevcut veriler arasındaki ilişkileri keşfedip yeni veriler için sonuç tahmin etme
* Geçmişteki verileri kullanarak veriye en uygun modeli bulma, yeni gelen verileri bu modele göre analiz edip sonuç üretme

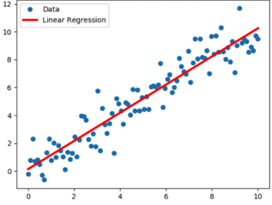
Makine öğrenmesinde 3 temel yöntem vardır (Şekil 1.3).



Şekil 1.3 Makine öğrenmesinin 3 temel yöntemi (Anonymous 2021c)

## Denetimli Öğrenme (ing. Supervised Learning)

Denetimli öğrenmede veriler etiketlidir (ing. labeled data), yani veri setindeki her bir verinin hangi sınıfa dahil olduğu ve yeni gelen bir verini dahil olacağı sınıflar bellidir. X girdi değerleri ile y çıktı değerleri arasındaki ilişki bulunur. Model, yeni X girdi değerlerine karşılık gelen y çıktı değerlerini tahmin edebilir. Örneğin Şekil 1.4’teki her bir nokta X ile y arasındaki ilişkiyi ifade eder. Noktaların geneline baktığımız zaman doğrusal bir eğilim gösterdiği açıktır. Yani bütün noktalar, başka bir ifade ile X ile y arasındaki ilişki, bir doğru ile temsil edilebilir. İşte bu doğrunun fit edilmesiyle X ile y arasındaki ilişki öğrenilmiş olur.

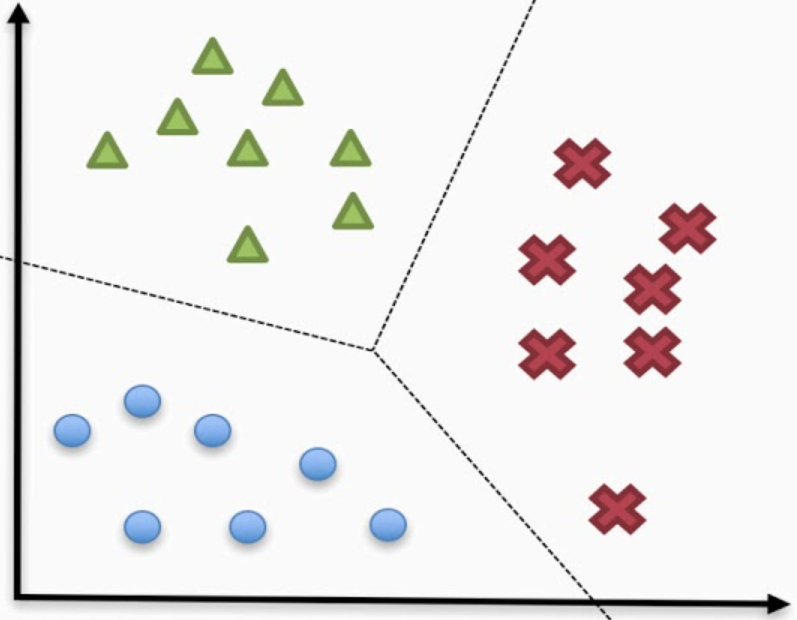


Şekil 1.4 X ile y arasındaki ilişki bir doğru ile temsil edilebilir

X ile y arasındaki ilişkinin bu şekilde bir fonksiyon ile temsil edilmesine regresyon (ing. regression) denilir. Nüfus artışı tahmini buna bir örnektir. Regresyon algoritmaları aşağıda verilmiştir:

* Lineer Regresyon (ing. Linear Regression)
* Lojistik Regresyon (ing. Logistic Regression)

X ile y arasındaki ilişki bir fonksiyon ile ifade edilemezse sınıflandırma (classification) yapılır (Şekil 1.5).



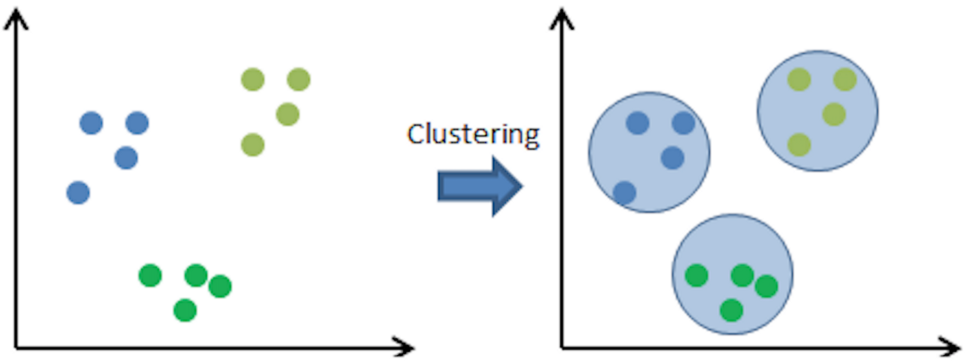
Şekil 1.5 Sınıflandırma

Örneğin bir e-posta sisteminde (Gmail, Hotmail gibi) bir e-postanın istenmeyen (ing. spam) olup olmadığının tespit edilmesi sorunu sınıflandırma ile çözülür. Geçmişte elde edilen bütün e-postalar 2 sınıfa ayrılır: normal e-posta veya istenmeyen e-posta. E-postalar bazı niteliklere sahiptir: postanın kimden geldiği, kimlere gönderildiği, konusunda ve içeriğinde hangi kelimelerin geçtiği. Bunların her birine öznitelik (ing. feature) denir. Geçmişte normal veya istenmeyen olduğu bilinen postalardan veri seti oluşturulur. Veri setindeki postaların öznitelikleri arasındaki benzerlikler, örüntüler, ilişkiler keşfedildiğinde normal ile istenmeyen e-posta farkı öğrenilmiş olur. Artık gelen yeni bir postanın özniteliklerine göre ilgili sınıfa (normal veya istenmeyen e-posta) dâhil edilir. Sınıflandırma algoritmaları aşağıda verilmiştir:

* Karar Destek Makineleri (ing. Support vector machines SVM)
* Bayes Sınıflandırıcı (ing. Bayes Classifier)
* Yapay Sinir Ağları (ing. Artificial Neural Networks)
* Genetik Algoritmalar (ing. Genetic Algorithms)
* Karar Ağaçları (ing. Decision Trees) – Rastgele Orman (ing. Random Forest)
* Örnek Tabanlı Yöntemler (ing. Instance Based Methods): k en yakın komşu (kNN)

## Denetimsiz Öğrenme (ing. Unsupervised Learning)

Denetimsiz öğrenmede X veri setindeki her bir veriye karşılık gelen y değerleri belirsizdir (ing. unlabeled data). Girdi değerleri arasındaki ilişkiler bulunarak birbiriyle ilişkili veriler kümelendirilir (Şekil 1.6). Yeni bir girdi bu kümelerden hangisiyle ilişkili ise o kümeye aittir.

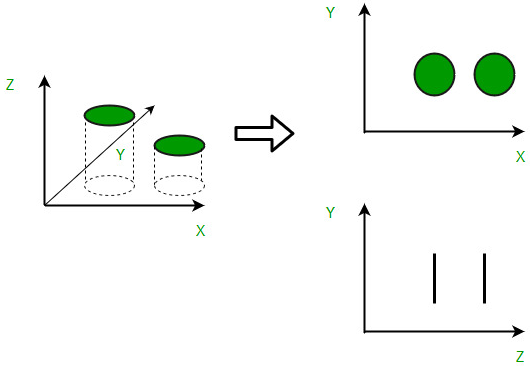


Şekil 1.6 X girdi veri setindeki veriler arasındaki ilişkilere göre kümeleme yapma

Örneğin bir e-ticaret sitesi müşterilerini gruplandırmak isteyebilir, görüldüğü üzere gruplar belli değildir yani y değerleri belirsizdir. Bu durumda neye göre gruplandırma yapılacağı merak konusu olabilir. Müşterilerin ne satın aldığına bakılmadan, yani X veri setindeki veriler arasındaki ilişkileri keşfetmeden, e-ticaret sitesinin neye göre kümeleyeceği bilinemez. Sonuçta yaşına, cinsiyetine, kaç kere alışveriş yaptığına veya herhangi başka bir kıstasa göre kümeleyebilir. Kümeleme algoritmaları aşağıda verilmiştir:

* K-Ortalama (ing. K-Means)
* Hiyerarşik
* Yapay Sinir Ağları
* Genetik Algoritmalar

Makine öğrenmesinde genel olarak çok büyük veri setleriyle çalışılır. Veri setinin boyutu modelin performansını, çalışma süresini etkiler. Bazı durumlarda veri setinin özniteliklerini azaltma yoluna gidilir. Buna boyut indirgeme (ing. dimentionality reduction) denir. Boyut indirgeme bazı öznitelikleri göz ardı etme (ing. feature elimination) veya mevcut özniteliklerden daha az sayıda öznitelik elde etme (ing. feature extraction) şeklinde olabilir. Örneğin Şekil 1.7’de 3 boyutta gösterilen silindirlerin z boyutu önemsiz görülüp göz ardı edilirse silindirler daireye dönüşür veya x boyutu göz ardı edilirse çizgiye dönüşür, böylece boyutu indirgenmiş olur.



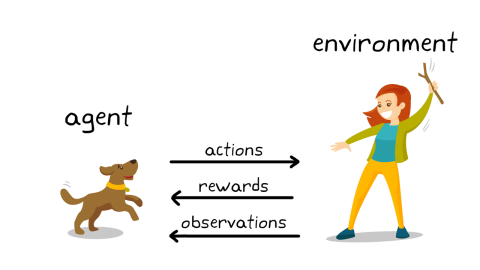
Şekil 1.7 Boyut indirgeme

Boyut indirgeme algoritmaları aşağıda verilmiştir:

* Temel Bileşen Analizi (ing. Principal Component Analysis PCA)
* Doğrusal Ayrıştırma Analizi (ing. Linear Discriminant Analysis LDA)
* Faktör Analizi (ing. Factor Analysis)
* Negatif olmayan matris çarpanlarına ayırma (ing. Non-negative Matrix Factorization NMF)

## Pekiştirmeli Öğrenme (ing. Reinforcement Learning)

Denetimli ve denetimsiz öğrenme yöntemlerinde öğrenme veri setindeki veriler arasındaki ilişkilerin bulunmasına dayalıyken pekiştirmeli öğrenmede giriş veri seti yoktur, çevrede gözlem yapılır ve seçenekler arasından en yüksek ödüllü olanı seçilir. Bu öğrenme yöntemi kısaca deneme/yanılma, ödül/ceza, neden/sonuç şeklinde tanımlanabilir. Şekil 1.8’de köpek öğrenme modelini, sahibi ve elindeki sopa çevreyi temsil etmektedir. Sahibinin attığı sopayı geri getiren köpek ödül kazanınca artık o davranışın kendi menfaatine olduğunu öğrenmiştir. Bundan sonra köpek atılan her sopayı geri getirecektir.



Şekil 1.8 Pekiştirmeli öğrenme (Anonymous 2021d)

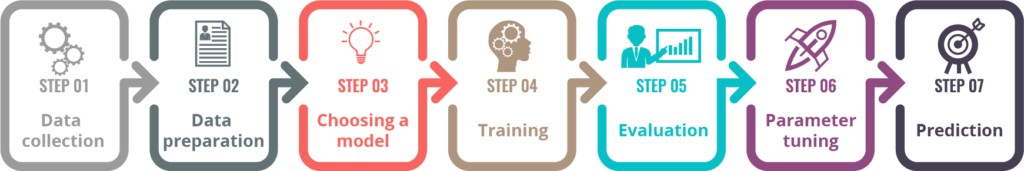
Bu öğrenme yöntemi robotlarda ve oyunlarda çok kullanılır. Arazide ilerleyen bir robot adımını arazinin o anki şartlarına göre atar. Satrançta yapılacak hamle taşların o anki dizilimine bağlıdır. IBM Deep Blue ve Google AlphaGo örnek olarak verilebilir. Pekiştirmeli öğrenme algoritmaları aşağıda verilmiştir:

* Markov Karar Süreci (ing. Markov Decision Process)
* Q Öğrenme (ing. Q Learning)

## Makine Öğrenmesi Süreci

Tipik bir makine öğrenmesi sürecinin (Şekil 1.9) adımları aşağıda açıklanmıştır:

* Makine öğrenmesini hangi probleme uygulayacaksak onunla ilgili veri toplamakla başlanır.
* Veriler hazır hale getirilir: boşluk içeren verileri ya mantıklı bir şekilde doldurma ya da o veriyi göz ardı etme, bir özniteliği normalize etme, ölçeklendirme vb.
* Çözmek istediğimiz problem için en uygun öğrenme modeli seçilir.
* Toplanan veri 3 kısma ayrılır: Eğitme (ing. training), doğrulama (ing. validation) ve test (ing. testing). Eğitim veri seti ile model eğitilir.
* Doğrulama veri seti ile model değerlendirilir, gerekirse öğrenme modeli değiştirilir veya modelin parametreleri hassaslaştırılarak model en iyi hale getirilir.
* Modelin o ana kadar hiç görmediği test veri seti ile model son kez test edilerek modelin öğrendiğinden emin olununca model artık tahmin yapmaya hazırdır.



Şekil 1.9 Makine öğrenmesi süreci (Anonymous 2021e)

## Yetersiz Öğrenme (ing. Underfitting) ve Aşırı Öğrenme (ing. Overfitting)

Yetersiz öğrenme modelin veriler arasındaki ilişkileri yeterince keşfedememesidir. Aşırı öğrenme ise modelin veriler arasındaki ilişkileri keşfetmek yerine verileri ezberlemesidir (Şekil 1.10). Aşırı öğrenme durumu yetersiz öğrenmeye nazaran daha çok görülür. Yetersiz öğrenme durumunda eğitim ve doğrulamada düşük doğruluk görülür. Aşırı öğrenme durumunda eğitim ve doğrulamada yüksek doğruluk ancak testte düşük doğruluk görülür.

Yetersiz öğrenme durumunu gidermek için aşağıdakiler yapılabilir:

* Modeli gözden geçirme
* Öznitelik sayısını arttırma
* Veriden gürültüleri giderme

Aşırı öğrenme durumunu gidermek için aşağıdakiler yapılabilir:

* Yeni veri ekleme
* Öznitelik sayısını azaltma
* Verileri düzenleme



Şekil 1.10 Yetersiz öğrenme ve aşırı öğrenme (Anonymous 2021f)

## Makine Öğrenmesi Kütüphaneleri

Makine öğrenmesi bir alana uygulanacağı zaman algoritmaların kodlanmasına çoğunlukla gerek yoktur; çünkü gönüllü yazılımcılar tarafından kütüphaneleri oluşturulmuştur. Bu kütüphaneler programa dâhil edilerek makine öğrenmesi algoritmalarına erişilebilir. Kütüphanelerin bazıları aşağıda verilmiştir:

* SciKit-Learn: Açık kaynak Python kütüphanesi (Anonymous 2021g)
* Google TensorFlow: Açık kaynak Python kütüphanesi (Anonymous 2021h)
* Keras: Derin öğrenme (ing. deep learning) kütüphanesi (Anonymous 2021i)
* Apache MXNet: Açık kaynak kütüphane C++ ve Python kütüphanesi (Anonymous 2021j)

Ayrıca makine öğrenmesini ticari hizmet olarak sunan kuruluşlar da vardır. IBM Watson, Microsoft Azure, Amazon SageMaker ücretli bulut hizmetlerinin bazılarıdır.

## Makine Öğrenmesi Veri Kaynakları

Makine öğrenmesinde pratik yapmak için veri setlerine ihtiyaç vardır. İnternette çok sayıda açık veri kaynakları mevcuttur. Bazıları aşağıda verilmiştir:

* <https://datasetsearch.research.google.com/>
* <https://www.kaggle.com/datasets>
* <https://computervisiononline.com/datasets>
* <http://deeplearning.net/datasets/>

Ayrıca bazı devletler kendi veri kaynaklarını oluşturmuşlardır. Avrupa Birliği, Hindistan, Avusturalya bazılarıdır. Web sayfalarından indirilip araştırmalarda kullanılmaya hazır veri setleri barındırırlar.

1. ASTRONOMİDE MAKİNE ÖĞRENMESİ UYGULAMALARINDAN ÖRNEKLER

Makine öğrenmesi astronomide çok sayıda alanda uygulanmıştır. Bazıları aşağıda verilmiştir:

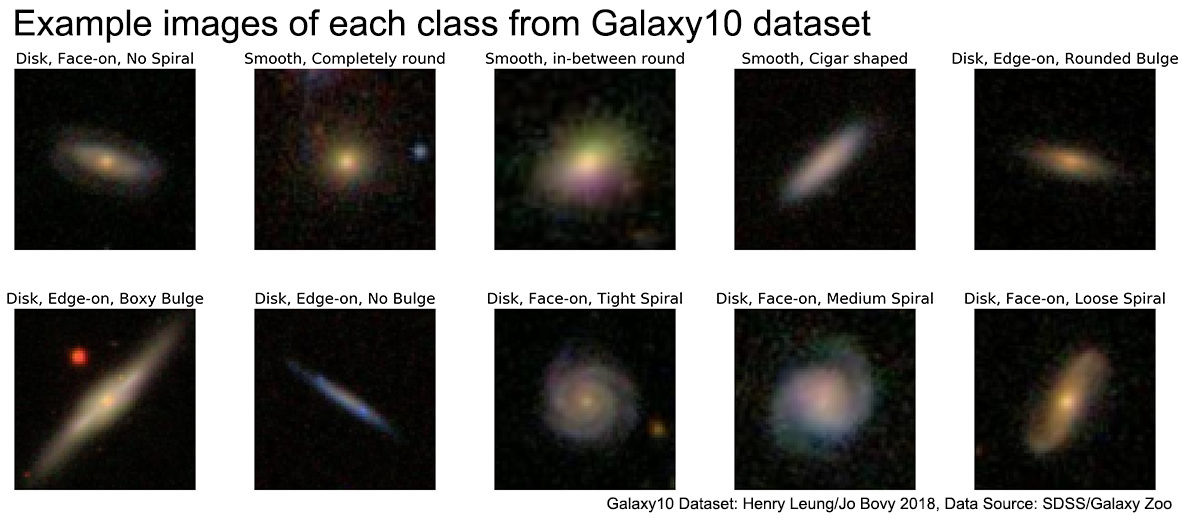
* Angel vd.’nin (1990) “Adaptive optics for array telescopes using neural-network techniques” çalışması makine öğrenmesinin astronomide uygulandığı ilk çalışmadır.
* Jenkins vd. (2012) Auto-Vetting adını verdikleri Rastgele Orman (ing. Random Forest) yöntemi ile Kepler Uzay Teleskobu ile elde edilen ışık eğrilerini kullanarak gezegen adaylarını gezegen olma olasılıklarına göre puanlamışlardır.
* Coughlin vd. (2016) Robovetter adını verdikleri Karar Ağaçları (ing. Decision Trees) yöntemi ile Kepler Uzay Teleskobu ile elde edilen ışık eğrilerini gezegen adayı geçişi veya Tip-I hata (ing. false-positive) şeklinde sınıflandırmışlardır.
* Shallue ve Vanderburg (2018) Astronet adını verdikleri Derin Evrişimsel Sinir Ağları (ing. Deep Convolutional Neural Network) yöntemi ile Kepler’den elde ettikleri ışık eğrilerini kullanarak eşik değeri aşan gözlemlerin (threshold crossing events, TCE) sınıflandırmasını yapmışlardır. Bu yöntemle Kepler 80’in 5., Kepler 90’ın 8. gezegeni doğrulandı. Kepler 90 Güneş sistemi dışında 8 gezegene sahip ilk sistemdir.
* Ansdell vd. (2018) Astronet’i temel alarak geliştirdikleri Exonet adını verdikleri bu çalışmada Kepler ışık eğrilerini “gezegen geçişi mi, değil mi?” şeklinde otomatik olarak sınıflandırıldı. Astronet’e nazaran daha iyi doğruluk elde edildi.
* Zucker ve Giryes (2018) TESS ışık eğrilerini kullanarak Güneş benzeri sistemlerde yaşanabilir bölgedeki gezegenlerin sığ geçişleri Evrişimsel Sinir Ağları (Convolutional Neural Network (CNN)) yöntemi ile tespit edildi. Modeli eğitmek için sentetik ışık eğrileri kullanıldı.
* Sturrock vd. (2019) tarafından Kepler Cumulative Object of Interest (KCOI) veri setindeki cisimler gezegen adayı veya Tip-I hata (ing. false-positive) şeklinde sınıflandırıldı. Rastgele Orman, kNN, SVM yöntemleri kullanıldı. Bu yöntemler karşılaştırıldı ve Rastgele Orman yönteminin en yüksek doğruluk (ing. accuracy) verdiği görüldü.
* Valizadegan vd. (2021) ExoMiner adını verdikleri bu çalışmada Kepler geçiş ışık eğrilerini Derin Öğrenme (ing. Deep Learning) yöntemiyle sınıflandırdılar. Bu şekilde 301 gezegen doğrulandı. Sınıflandırmada önceki çalışmalara nazaran en iyi doğruluk elde edildi.
* Ofman vd. (2021) TESS ışık eğrilerini kullanılarak TCE’leri sınıflandırdı. Modeli eğitmek için Kepler ışık eğrileri kullanıldı. 3 yeni gezegen adayı bulundu.
* Pratyush ve Gangrade (2021) Kepler ışık eğrilerinde muhtemel gezegen geçişlerini tespit etmek için Stacked GBDT (Gradient Boosting Decision Trees) yöntemini önerdi. Bu yöntem geleneksel yöntemlere nazaran daha iyi performans gösterdi. Ardından gezegen geçişleri, aday veya Tip-I Hata (false-positive) şeklinde sınıflandırıldı. Aday olanlar için ise sınıflandırma ve regresyon yöntemleri kullanılarak yaşanabilirlik değerlendirmesi (habitability assessment) yapıldı.

1. ÖRNEK MAKİNE ÖĞRENMESİ UYGULAMASI

Python dilinde yazılan bu örnek uygulamaya Anonymous (2021k) adresinden ulaşılabilir. Kodun yazımında aşağıdaki kaynaklardan yararlanıldı:

* <https://astronn.readthedocs.io/en/latest/galaxy10.html>
* <https://www.tensorflow.org/tutorials/keras/classification>
* <https://towardsdatascience.com/a-simple-cnn-multi-image-classifier-31c463324fa>
* <https://www.geeksforgeeks.org/python-image-classification-using-keras/>

Bu uygulamada Sinir ağları (ing. Neural Network) yönteminin özel bir hali olan Derin Öğrenme (ing. Deep Learning) yöntemiyle galaksi görüntüleri sınıflandırıldı. Veri seti olarak Anonymous (2021l) kaynağındaki 10 sınıfa ayrılmış 21785 imaj kullanıldı (Şekil 3.11).

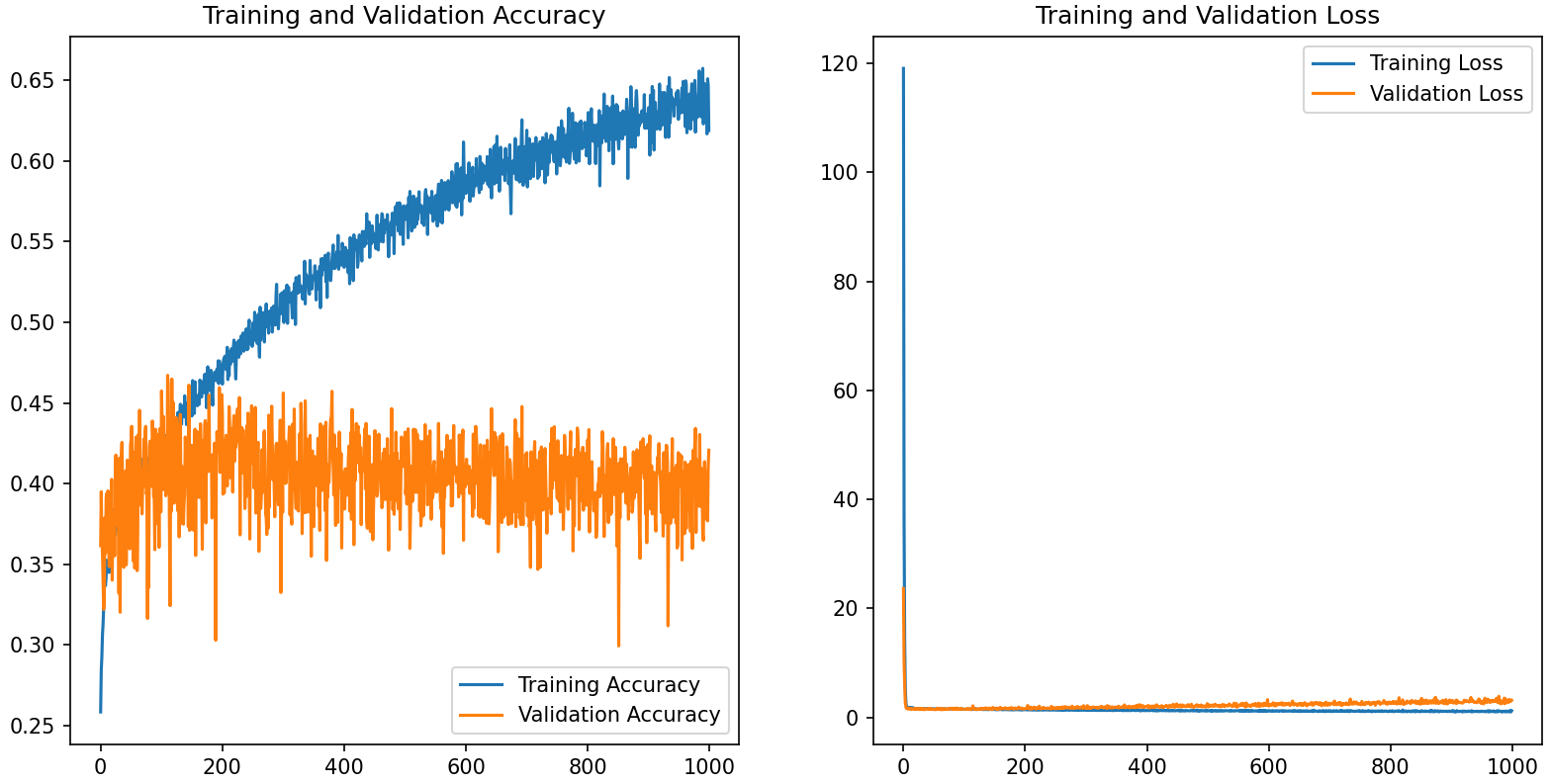


Şekil 3.11 10 sınıfa ayrılmış galaksi imajlarından örnekler (Anonymous 2021m)

21785 imaj 3 sete ayrıldı:

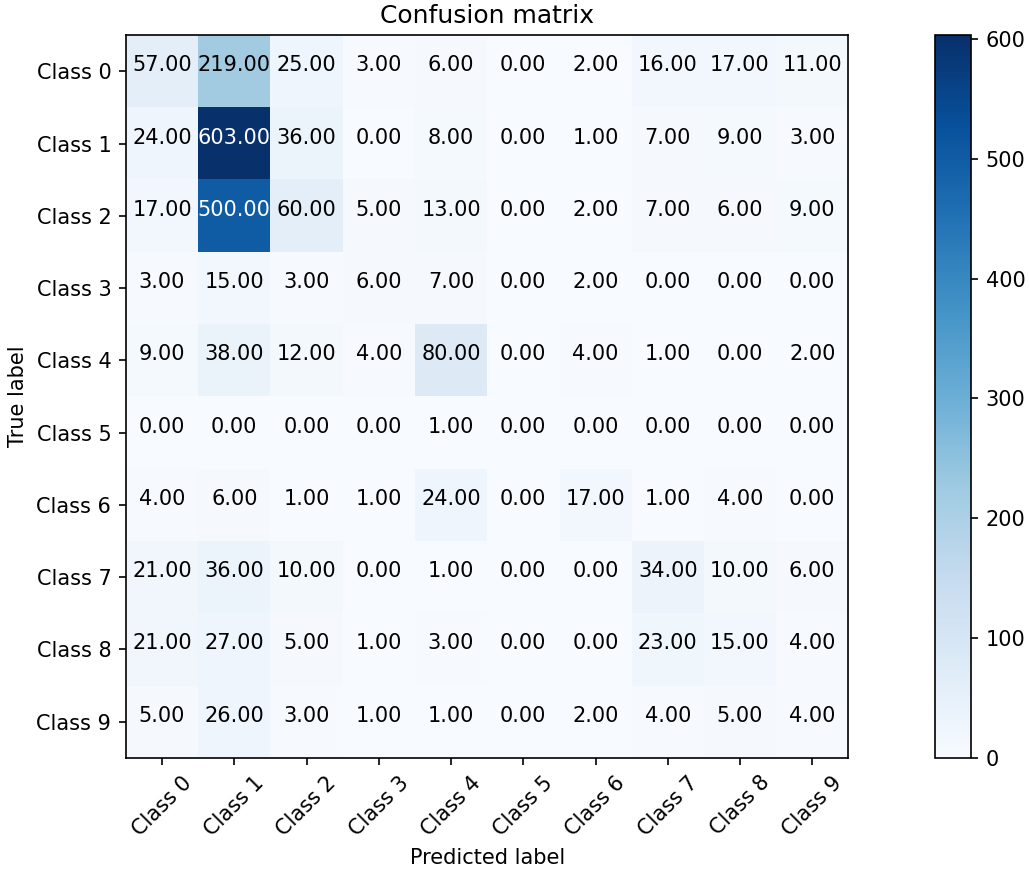
* %72 Training: 15684
* %18 Validation: 3922
* %10 Test: 2179

Kod içerisinde (detaylar için Anonymous (2021k) adresine bakınız) epoch=1000 parametresiyle öğrenme modelinin öğrenme sürecini 1000 kez tekrarlaması sağlandı (Şekil 3.12). Bu parametrenin artırılması doğruluk (ing. accuracy) oranını artıracaktır.



Şekil 3.12 1000 çevrim ile elde edilen Doğruluk (ing. Accuracy) ve Kayıp (ing. Loss) grafikleri

Elde edilen Doğruluk ve Kayıp grafikleri aslına tatmin edici değildir. Modelin parametrelerinin hassaslaştırılması hatta modelin değiştirilmesi gerekebilir. Bu uygulama makine öğrenmesine bir örnek olduğu için modelin yeterli öğrendiği varsayıldı. Son olarak test verisi ile model test edilerek hata matrisi elde edildi (Şekil 3.13).



Şekil 3.13 Hata matrisi (ing. confusion matrix)

Hata matrisi öğrenme modelinin tahminleri ile verilerin gerçekte ne olduklarını karşılaştırır. Yatay eksen tahminleri, düşey eksen gerçek sınıfları temsil etmektedir. Matristeki her bir sayı modelin tahmin sayısını vermektedir. Örneğin Class 9 (Sınıf 9) ile Class 0’a denk gelen 5 sayısı, gerçekte Class 9’da olan 5 imajı model yanlış olarak Class 0 olarak sınıflandırmıştır. Modelin doğru olarak sınıflandırdıkları yatayda ve düşeyde aynı sınıfa ait olan sayılardır. Örneğin Class 0 ile Class 0’a denk gelen 57 sayısı model Class 0’a ait olan 57 imajı doğru olarak sınıflandırmıştır. Dolayısıyla matrisin sol-üstten sağ-alta köşegeni oluşturan sayılar modelin doğru tahmin ettiklerini, diğer sayılar modelin yanlış tahmin ettiklerini ifade etmektedir. Yani ideal bir hata matrisinde köşegendeki sayıların maksimum, diğer sayıların minimum olması gerekmektedir.

KAYNAKLAR

Anonymous. 2021a. Web Sitesi: <https://ibm.com>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021b. Web Sitesi: <https://wikipedia.org>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021c. Web Sitesi: <https://i.pinimg.com/originals/3e/21/37/3e2137475e54d82f3cbb45779bdb6b47.png>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021d. Web Sitesi: <https://www.mathworks.com/discovery/reinforcement-learning.html>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021e. Web Sitesi: <https://medium.com/dataseries/7-steps-to-machine-learning-how-to-prepare-for-an-automated-future-78c7918cb35d>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021f. Web Sitesi: <https://www.kdnuggets.com/2018/04/data-science-interview-guide.html/2>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021g. Web Sitesi: <https://github.com/scikit-learn/scikit-learn>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021h. Web Sitesi: <https://github.com/tensorflow/tensorflow>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021i. Web Sitesi: <https://github.com/keras-team/keras>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021j. Web Sitesi: <https://github.com/apache/incubator-mxnet>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021k. Web Sitesi: <https://github.com/BedriKeskin/Galaxy-Classification>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021l. Web Sitesi: <http://astro.utoronto.ca/~bovy/Galaxy10/Galaxy10.h5>. Erişim Tarihi: 11.12.2021.

Anonymous. 2021m. Web Sitesi: <https://astronn.readthedocs.io/en/latest/galaxy10.html>. Erişim Tarihi: 11.12.2021.

Angel, J. vd.  1990. Adaptive optics for array telescopes using neural-network techniques. Nature 348, 221–224.

Ansdell vd. 2018. Scientific Domain Knowledge Improves Exoplanet Transit Classification with Deep Learning. The Astrophysical Journal Letters, Volume 869, Issue 1, article id. L7, pp. 7.

Graff vd. 2014. SKYNET: an efficient and robust neural network training tool for machine learning in astronomy. MNRAS 441, 1741–1759.

Ofman vd. 2021. Automated identification of transiting exoplanet candidates in NASA Transiting Exoplanets Survey Satellite (TESS) data with machine learning methods. New Astronomy, Volume 91, article id. 101693.

Pratyush ve Gangrade. 2021. Automation Of Transiting Exoplanet Detection, Identification and Habitability Assessment Using Machine Learning Approaches. arXiv preprint arXiv:2112.03298.

Shallue, C. J. ve Vanderburg, A. 2018. Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90. The Astronomical Journal, 155(2), 94.

Sturrock, G. C. vd. 2019. Machine Learning Pipeline for Exoplanet Classification. SMU Data Science Review: Vol. 2: No. 1, Article 9.

Turing, A. 1950. Computing Machinery and Intelligence. Volume LIX, Issue 236.

Vanderplas vd. 2012. Introduction to astroML: Machine learning for astrophysics, proc. of CIDU, pp. 47-54.

Valizadegan vd. 2021. ExoMiner: A Highly Accurate and Explainable Deep Learning Classifier that Validates 301 New Exoplanets. Bulletin of the American Astronomical Society, Vol. 53, No. 6.

Zucker ve Giryes. 2018. Shallow Transits—Deep Learning. I. Feasibility Study of Deep Learning to Detect Periodic Transits of Exoplanets. The Astronomical Journal, 155:147.