**800100807140 SEMİNER**

**2021-2022 Güz Dönemi**

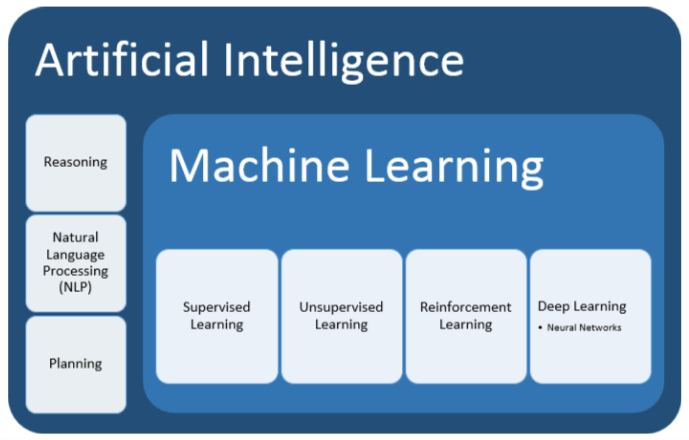
**Bedri KESKİN 19820101**

**Danışman: Doç. Dr. Özgür BAŞTÜRK**

**ASTRONOMİDE MAKİNE ÖĞRENMESİ UYGULAMALARI**

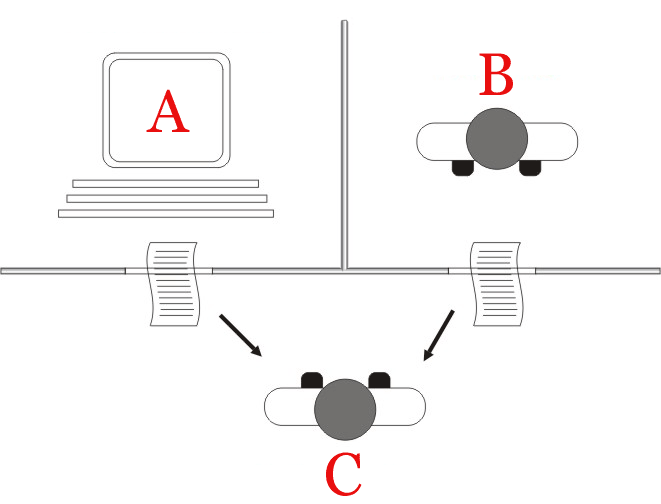
1. **Makine Öğrenmesi**

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



Şekil 1.1. Yapay Zeka / Makine Öğrenmesi İlişkisi

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, Leibnez’in 4 işlem yapabilen hesap makinesi düşünen makinelerin öncüleri olarak düşünülebilir. Yapay zeka 1950’lerde ortaya çıkmıştır. Alan Turing’in “Computing Machinery and Intelligence” makalesi bir dönüm noktasıdır. Bu makalesinde Turing Testi olarak bilinen bir yapay zekanın başarılı olmasının ölçütünü ortaya koymuştur. Bir yapay zeka sistemine ve bir insana sorular sorulur (Şekil 1.2). Verilen cevaplarda hangisinin yapay zeka hangisinin insan olduğu ayırt edilemezse o yapay zeka sistemi başarılıdır.



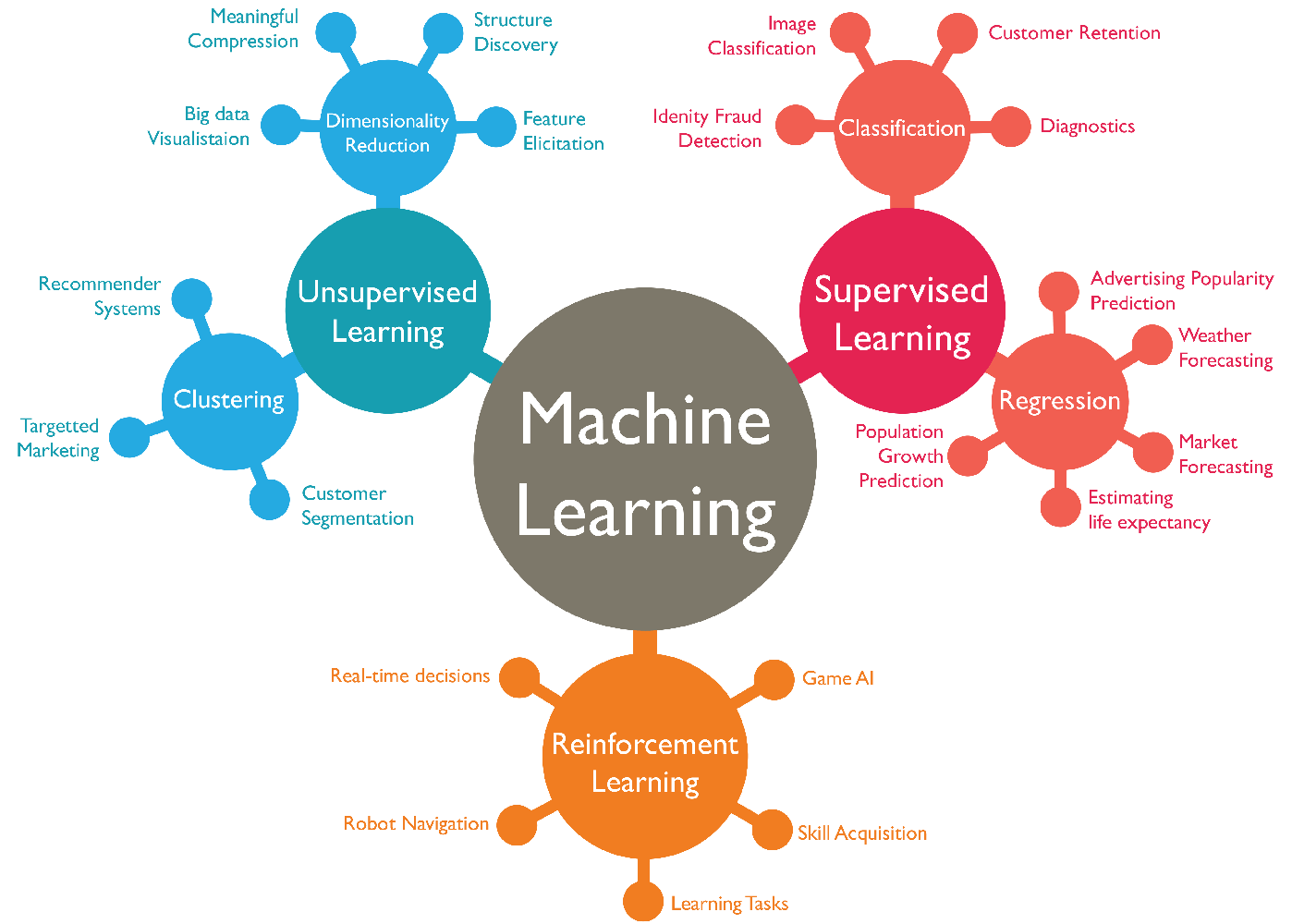
Şekil 1.2. Turing testi (wikipedia.org).

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

Makine öğrenmesinde temel prensip aşağıdaki şekillerde 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

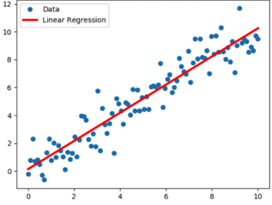
Makine öğrenmesinde 3 temel yöntem vardır (Şekil 1.3).



Şekil 1.3. Makine öğrenmesinin 3 temel yöntemi. (https://i.pinimg.com/originals/3e/21/37/3e2137475e54d82f3cbb45779bdb6b47.png)

* 1. **Denetimli Öğrenme (Supervised Learning)**

Denetimli öğrenmede veriler etiketlidir (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.

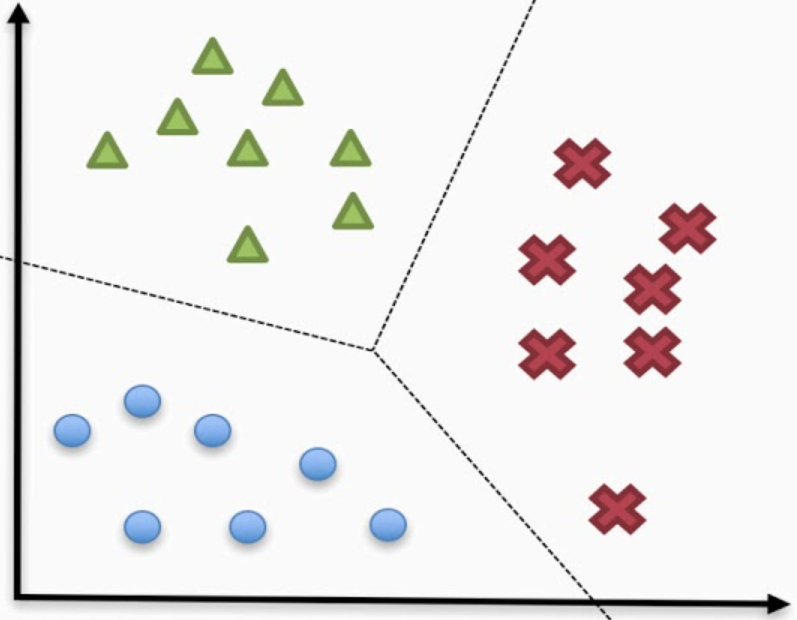


Şekil 1.4. X ile y arasındaki ilişki bir doğru ile temsil edilebilir

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

* Linear Regression
* Logistic Regression

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

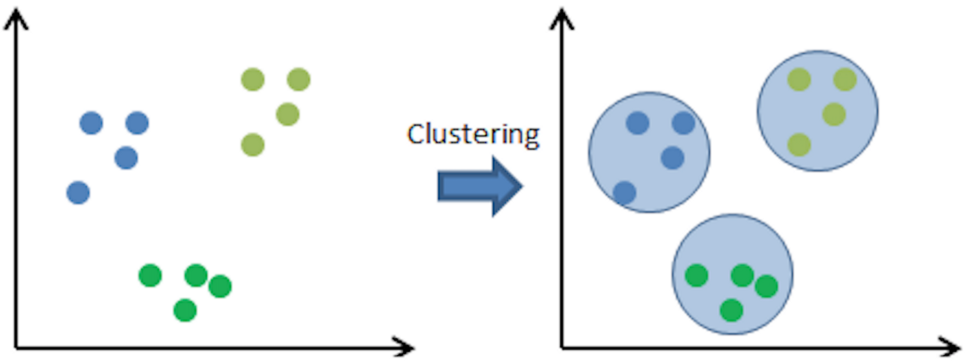


Şekil 1.5. Sınıflandırma

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

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

Denetimsiz öğrenmede X veri setindeki her bir veriye karşılık gelen y değerleri belirsizdir (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.

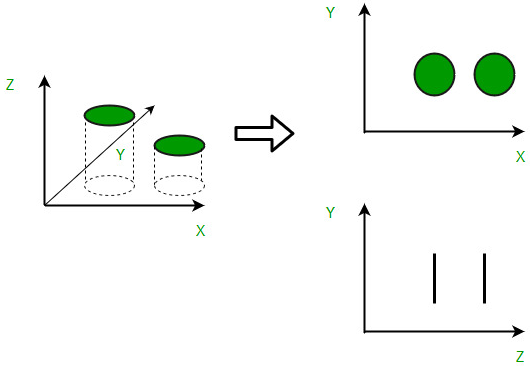


Şekil 1.6. X girdi veri setindeki veriler arasındaki ilişkilere göre kümeleme yapılır.

Örneğin bir e-ticaret sitesi müşterilerini gruplandırmak isteyebilir, görüldüğü üzere gruplar belli değildir yani y değerleri belirsizdir fakat neye göre gruplandıracak? Müşterilerin ne satın aldığına bakmadan yani X veri setindeki veriler arasındaki ilişkileri keşfetmeden neye göre kümeleyeceğini bilemez. Sonuçta yaşına, cinsiyetine, kaç kere alışveriş yaptığına veya herhangi başka bir kıstasa göre kümeleyebilir. Kümeleme algoritmaları aşağıda verilmiştir:

* 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 (dimentionalty reduction) denir. Boyut indirgeme bazı öznitelikleri göz ardı etme (feature elimination) veya mevcut özniteliklerden daha az sayıda öznitelik elde etme (feature extraction) şeklinde olabilir. Örneğin Şekil 1.7’de 3 boyutta gösterilen silindirlerin z boyutu önemsiz görülüp göz ardı edilirse silidirler daireye dönüşür veya x boyutu göz ardı edilirse çizgiye dönüşür, böylece boyutu indirgenmiş olur.

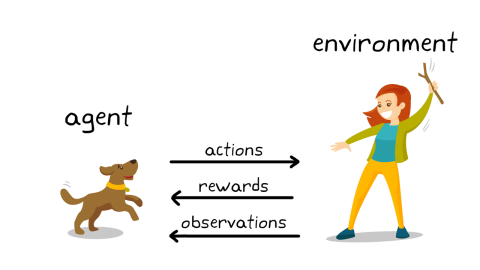


Şekil 1.7. Boyut indirgeme.

Boyut indirgeme algoritmaları aşağıda verilmiştir:

* Temel Bileşen Analizi (Principal Component Analysis PCA)
* Doğrusal Ayrıştırma Analizi (Linear Discriminant Analysis LDA)
* Faktör Analizi (Factor Analysis)
* Negatif olmayan matris çarpanlarına ayırma (Non-negative matrix factorization NMF)
  1. **Pekiştirmeli Öğrenme (Reinforcement Learning)**

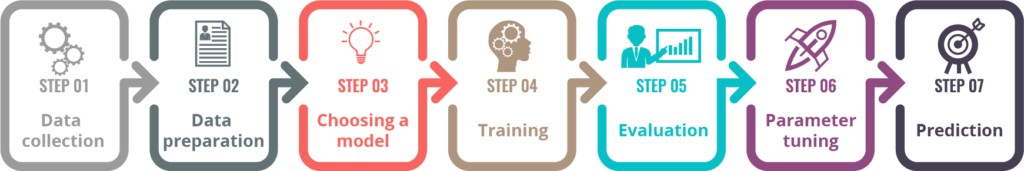
Denetimli ve denetimsiz öğrenme yöntemlerinde öğrenme veri setindeki veriler arasındaki ilişkilerin bulunmasına dayalıyken pekiştirmeli öğrenmede giriş veri seti yoktur, çevrede gözlem yapılır ve seçenekler arasından en yüksek ödüllü olanı seçilir. Bu öğrenme yöntemi kısaca deneme/yanılma, ödül/ceza, 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.

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 Decision Process
* Q learning
  1. **Makine Öğrenmesi Süreci (Şekil 1.9)**
* 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 gözardı etme, bir özniteliği normalize veya scale etme vb.
* Çözmek istediğimiz problem için en uygun öğrenme modeli seçilir
* Toplanan veri 3 kısma ayrılır: Training, validation ve test. Training veri seti ile model eğitilir.
* Validation veri seti ile model değerlendirilir, gerekirse öğrenme modeli değiştirilir, 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 olunca model artık tahmin yapmaya hazırdır.



Şekil 1.9. Makine öğrenmesi süreci

* 1. **Yetersiz Öğrenme (Underfitting) ve Aşırı Öğrenme (Overfitting):**

Yetersiz öğrenme modelin veriler arasındaki ilişkileri yeterince keşfedememesidir, aşırı öğrenme modelin veriler arasındaki ilişkileri keşfetmek yerine verileri ezberlemesidir (Şekil 1.10). Overfit durumu underfite nazaran daha çok görülür. Underfit durumunda eğitim ve doğrulamada düşük doğruluk görülür. Overfit durumunda eğitim ve doğrulamada yüksek doğruluk ancak testte düşük doğruluk görülür.

Yetersiz öğrenme durumunu gidermek için:

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

Aşırı öğrenme durumunu gidermek için:

* Yeni veri ekleme
* Öznitelik sayısını azaltma
* Verileri düzenleme



Şekil 1.10. Yetersiz öğrenme ve aşırı öğrenme

* 1. **Makine Öğrenmesi Kütüphaneleri**

Makine öğrenmesini bir alana uygulanacağı zaman algoritmaları kodlamamıza gerek yoktur çünkü gönüllü yazılımcılar tarafından kütüphaneleri oluşturulmuştur. Bu kütüphaneleri programımıza dâhil ederek makine öğrenmesi algoritmalarına erişebiliriz. Kütüphanelerin bazıları aşağıda verilmiştir:

* SciKit-Learn: Açık kaynak Python kütüphanesi <https://github.com/scikit-learn/scikit-learn>
* Google TensorFlow: Açık kaynak Python kütüphanesi <https://github.com/tensorflow/tensorflow>
* Keras: Derin öğrenme (deep learning) kütüphanesi <https://github.com/keras-team/keras>
* Apache MXNet: Açık kaynak kütüphane. Apache Software Foundation. C++ ve Python. <https://github.com/apache/incubator-mxnet>

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

* 1. **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. **Astronomiye Yönelik Makine Öğrenmesi Uygulamalarından Örnekler**

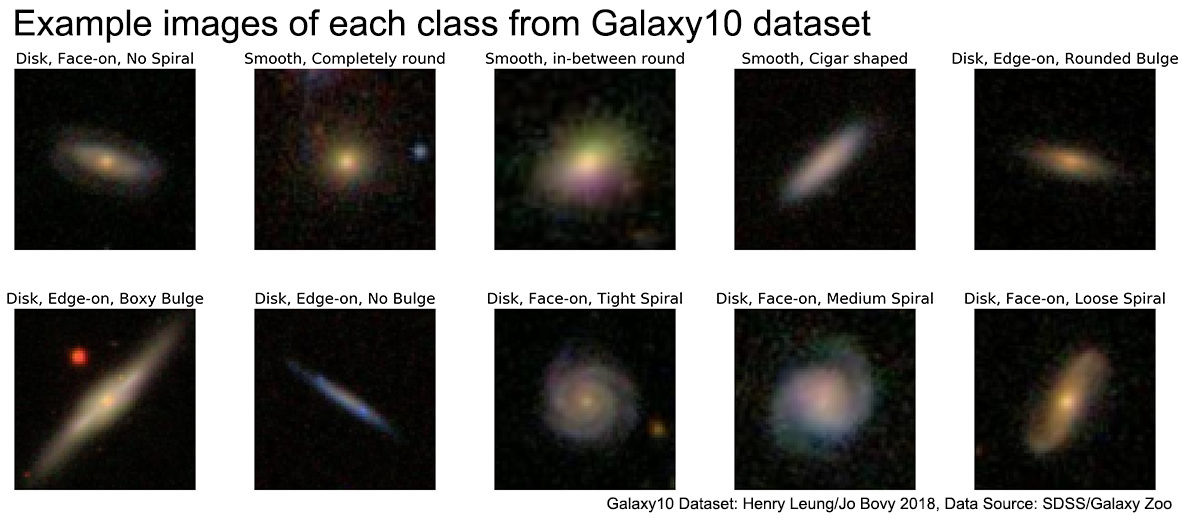
* Angel vd.’nin (1990) “Adaptive optics for array telescopes using neural-network techniques” çalışması makine öğrenmesinin astronomide uygulandığı ilk çalışmadır.
* Jenkins vd. (2012) Auto-Vetting adını verdikleri, Kepler’den elde ettikleri ışık eğrilerini kullanarak Random Forest yöntemi ile gezegen adaylarını gezegen olma olasılıklarına göre puanladılar.
* Coughlin vd. (2016) Robovetter adını verdikleri, Kepler’den elde ettikleri ışık eğrileri Decision Trees yöntemi ile gezegen adayı veya false-positive (yanlış-doğru) şeklinde sınıflandırıldı.
* Shallue ve Vanderburg (2018) Astronet adını verdikleri, Kepler’den elde ettikleri ışık eğrilerini kullanarak Deep convolutional neural network yöntemi ile TCE’lerin (treshold crossing events) sınıflandırılması yapıldı. 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 geliştirdiler. Exonet adını verdikleri bu çalışmada Kepler transit verisi gezegen mi değil mi şeklinde otomatik olarak sınıflandırıldı. Astronet’e nazaran daha iyi doğruluk elde edildi.
* Sturrock vd. (2019) tarafından Kepler Cumulative Object of Interest (KCOI) veri seti gezegen adayı veya false-positive (yanlış-doğru) şeklinde sınıflandırıldı. Random Forest, kNN, SVM yöntemleri kullanıldı. Bu yöntemler karşılaştırıldı ve Random Forest yönteminin en yüksek doğruluk (accuracy) verdiği görüldü.
* Valizadegan vd. (2021) ExoMiner adını verdikleri bu çalışmada Kepler transit verisi Deep Learning yöntemiyle sınıflandırıldı. 301 gezegen doğrulandı. Sınıflandırmada önceki çalışmalara nazaran en iyi doğruluk elde edildi.

1. **Örnek Makine Öğrenmesi Uygulaması**

Python dilinde yazılan bu örnek uygulamaya <https://github.com/BedriKeskin/Galaxy-Classification> 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 Neural Network yönteminin özel bir hali olan Deep Learning yöntemiyle galaksi görüntüleri sınıflandırıldı. Veri seti olarak <http://astro.utoronto.ca/~bovy/Galaxy10/Galaxy10.h5> adresindeki 10 sınıfa ayrılmış 21785 imaj kullanıldı (Şekil 3.1).

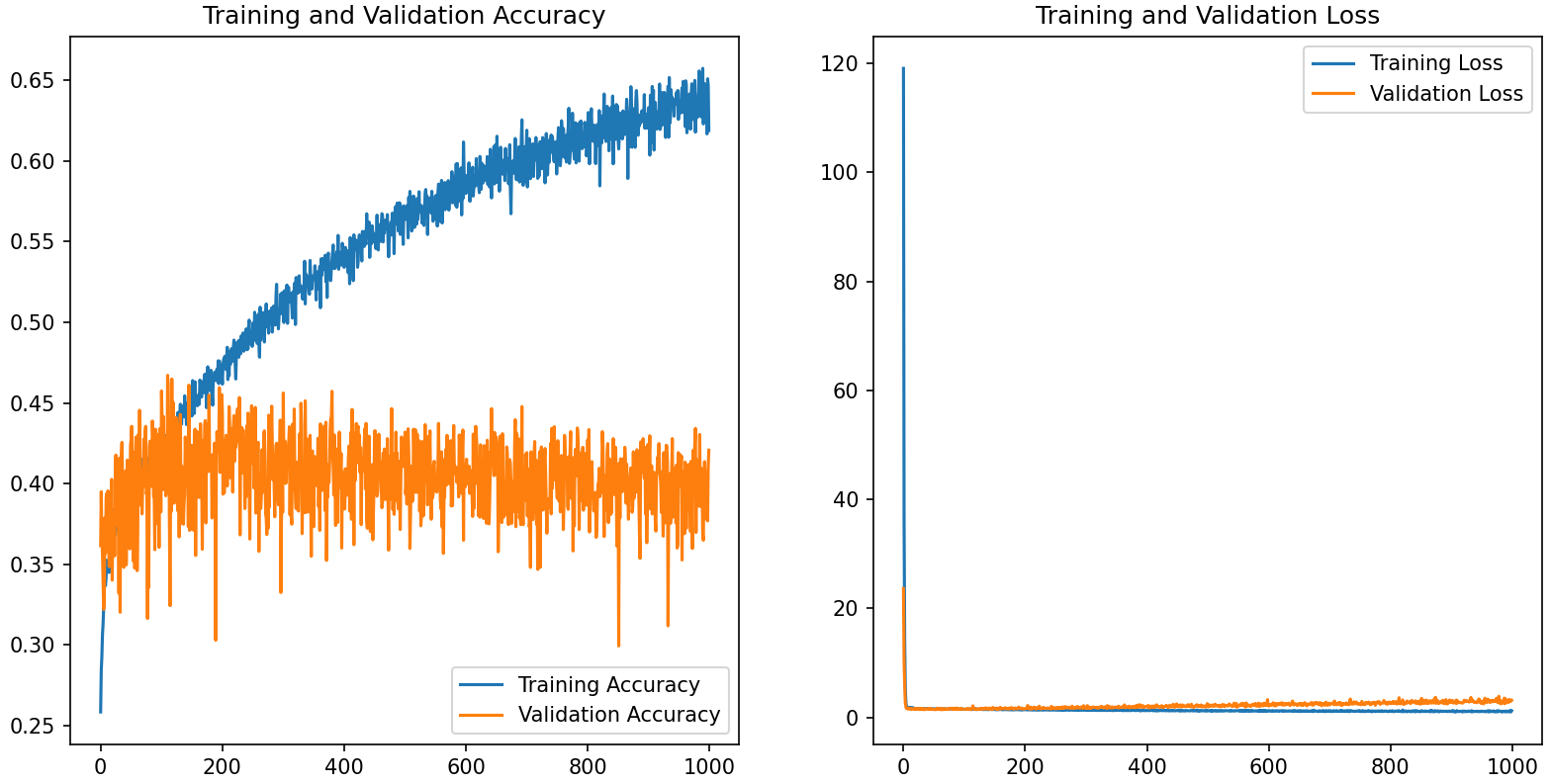


Şekil 3.1. 10 sınıfa ayrılmış galaksi imajlarından örnekler.

21785 imaj 3 sete ayrıldı:

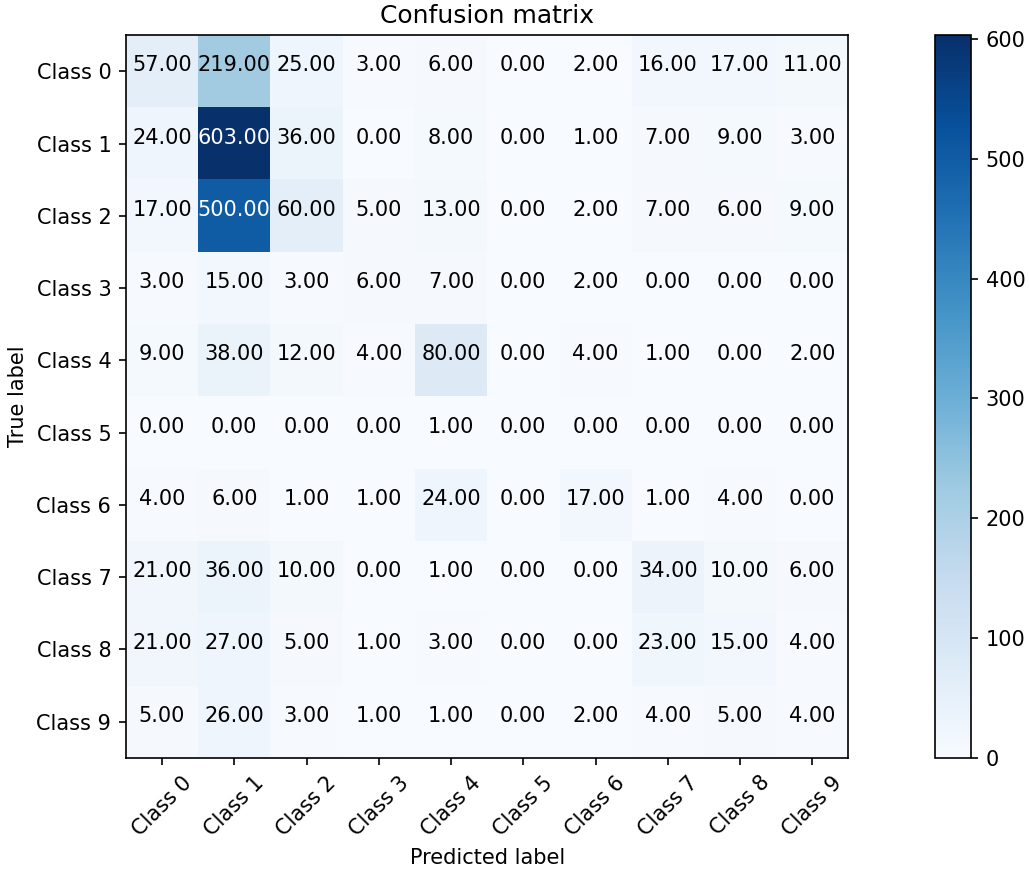
* + %72 Training: 15684
  + %18 Validation: 3922
  + %10 Test: 2179

Kod içerisinde (detaylar için Github linkine bakınız) epoch=1000 parametresiyle öğrenme modelinin öğrenme sürecini 1000 kez tekrarlaması sağlandı (Şekil 3.2). Bu parametrenin artırılması doğruluk (accuracy) oranını artıracaktır ancak modeli aşırı öğrenme durumuna sokabilir.



Şekil 3.2. 1000 çevrim sonunda elde edilen Doğruluk (Accuracy) ve Kayıp (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 edildi (Şekil 3.3).



Şekil 3.3. Hata matrisi (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 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:**

* 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.
* 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.
* 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.