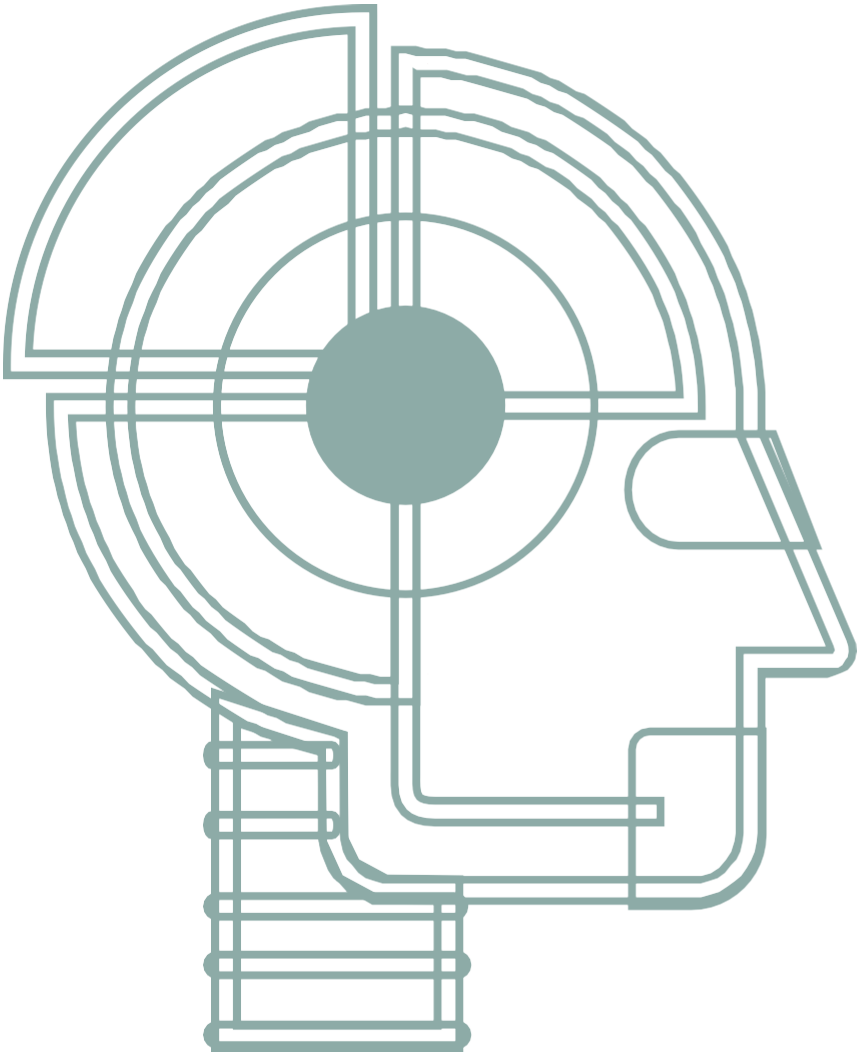
#MİLLİ TEKNOLOJİ HAMLESİ

DENEYAP TÜRKİYE

YAPAY ZEKÂ

ORTAOKUL



TÜBİTAK

TURKİYE TEKNOLOJİ TAKİMİ

TÜBİTAK Deneyap Kitapları 13

Yapay Zekâ

ORTAOKUL

Doç. Dr. Utku KÖSE
Doç. Dr. Koray ÖZSOY
Dr. Öğr. Üyesi Bekir AKSOY

TÜBİTAK Deneyap Kitapları *DENEYAP TÜRKİYE* Projesi kapsamında hazırlanmıştır.

ISBN: 978-605-312-517-4
Yayıncı Sertifika No: 47703

Yayım Tarihi: Eylül 2023

TÜBİTAK Başkanı: Prof. Dr. Hasan MANDAL
Bilim ve Toplum Başkanı: Ömer KÖKÇAM
Genel Yayın Yönetmeni: Fatma BAŞAR
Editörler: Dr. İpek PİRPİROĞLU GENCER - Havva Hilal KAÇAR
Düzelti: Dr. Mustafa ORHAN
Telif İşleri Sorumlusu: Havva Hilal KAÇAR

TÜBİTAK Bilim ve Toplum Başkanlığı
Tunus Caddesi No: 80 Kavaklıdere 06680 Ankara
Tel: (312) 298 96 50
e-posta: deneyap@tubitak.gov.tr
https://yayinlar.tubitak.gov.tr/deneyap-atolyesi

DENEYAP
Teknoloji Atölyeleri

YAPAY ZEKÂ

ORTAOKUL

Doç. Dr. Utku KÖSE
Doç. Dr. Koray ÖZSOY
Dr. Öğr. Üyesi Bekir AKSOY

TÜBİTAK

İçindekiler

Sunuş

Tüm dünyada olduğu gibi ülkemizde de teknolojik gelişmeler diğer bir ifade ile dijital endüstri hızlı biçimde değişmektedir. Endüstri başta olmak üzere ekonominin sağlık, havacılık-savunma sanayi, otomotiv, imalat, tarım ve eğitim başta olmak üzere tüm alanlarında dijital dönüşümü gerçekleştirmek için; Büyük Veri, Sensörler/Algılama Sistemleri, Nesnelerin İnterneti, Artırılmış Gerçeklik, Bulut Teknolojileri, Siber Güvenlik, Akıllı Fabrikalar, Eklemeli imalat, Yapay Zekâ gibi teknolojiler kullanılmaktadır. İnternet teknolojilerin hızla gelişmesi ile üretilen büyük verilerin hacimleri, geliştirilen algoritmalar ve veri depolama sistemlerindeki gelişmeler yapay zekâ teknolojilerini bugün çok daha popüler hale getirmiştir. Bu nedenle ülkemizdeki öğrencilerin teknolojik gelişmelere entegrasyonunda ve milli teknolojik hamlelerin geliştirilmesinde en önemli uygulamalı eğitim kurumlarından birisi de TÜBİTAK Deneyap Teknoloji Atölyeleri'dir.

Yapay zekâ teknolojileri, öğrencilerin bilişsel ve analitik düşünme becerisini geliştirmek için uygun bir teknolojik yöntemdir. Bu yöntem uygun bir eğitmen entegrasyonuyla gerçekleştirildiğinde öğrencilerin analitik ve zihinsel düşünme ile mesleki ve toplumsal gelişimlerine üst düzey problem çözme, yaratıcılık becerilerini geliştirmesine katkı sağlayacağı düşünülmektedir. Hazırlanan kitapta öğrenci ve eğitmenlere yapay zekâ teknolojileri program geliştirme ve uygulama yapma sürecinde rehberlik etmesi hedeflenmiştir. Öğretmenlerin öğrencilerin gelişmesine yardımcı olabilmesi için adım adım kod yazma öğretim modeli kullanmıştır. Bu amaç için bu kitabın her aşamasında yazarlar tarafından "Algıla, Tasarla, Harekete Geç, Yürüt ve Karar Ver" öğrenme döngüsü geliştirilmiştir.

Öğrenme döngüsünde yapay zekâ teknolojileri algoritmalarının temel kavramların oluşturulmasında eğitmen temelli algıla yaklaşımı sunulmuştur. Öğrencilerin verilen temel yapay zekâ bileşenleri ile ileri seviye dikkat ve motivasyonlarını sağlama yaklaşımları ön plana çıkarılmaya çalışılmıştır. Tasarla bölümünde eğitmen yapay zekâ veri setini öğretme ve hazırlama işlemlerini gerçekleştirmektedir. Hareket geç bölümünde eğitmen öğrencilerden Tasarla bölümünde öğretilen veri setlerine uygun yapay zekâ yöntemlerine ait kodları birlikte yazar. Öğrenciler Yürüt bölümünde Tasarla ve Harekete Geç aşamalarında hazırlamış oldukları uygulamalar için yapay zekâ modellerine ait eğitim ve tahminlenmeden elde edilecek sonuçları doğru biçimde kodlaması sağlar. Karar ver bölümünde yapay zekâ modelinden elde edilen sonuçların başarısını test eder. Bu öğretim döngüsünden sonra eğitmen öğrencilere her haftaya ait birer etkinlik uygulaması vererek öğrencilerin yapay zekâ algoritmalarına yönelik Python programlama dili kullanarak kod yazma, hata ayıklama, modeli eğitme, tahminleme ve modelden elde edilecek doğruluk başarım sonuçlarını test eder.

Kitapta yapay zekâ algoritmalarının modellenmesi için kolay öğrenebilen, nesne tabanlı uygulamaları ile oldukça sıklıkla tercih edilen açık kaynak kodlu Python programlama dilindeki Spyder editörü kullanılmıştır. Yapay zekâ modellerini uygulamak için Python programlama dilinde sıklıkla kullanılan Numpy, Matplotlib, Scipy, Scikit-Learn, TensorFlow, Keras, Pytorch kütüphaneleri kullanılmıştır. Ayrıca her hafta öğrencilerin yapay zekâ modellerini daha iyi anlayabilmeleri için açık erişimli internet sitelerinden farklı konularda farklı veri setleri kullanılarak uygulamaları yapması sağlanmıştır. Eğitmen etkinlik öncesinde öğrencilere yapay zekâ modelleri kullanılarak gerçekleştirilecek uygulamaları için açık erişimli internet sitesinden veri setini yükleyerek öğrenciler tarafından kullanılmasını sağlar. Bu kitap ortaokul seviyesinde yapay zekâ eğitimi vermek isteyen eğitmenler için uygulamalı biçimde hazırlanmıştır. Kitapta, öğrenciler ilk aşamada yapay zekanın kuramsal temellerini, ikinci aşamada problemin belirleyerek açık erişimli internet sitelerinden veri setlerini yüklenmesi ve verilerin analizi adımlarını ve son aşamada ise yapay zekâ modelleri kullanılarak problemi çözme, tahminleme ve raporlama adımlarını kazanması amaçlanmıştır. Konuların tamamı öğrenme döngüsü kavramı içerisinde oluşturulmuştur. Kitap 8 haftalık bir ders planı olarak tasarlanmıştır. Her haftaya bir bölüm düşecek şekilde sekiz haftalık içerik hazırlanmıştır. Öğrencilerin öğrenmiş oldukları yapay zekâ modelleri ile ilgili 4. haftadan itibaren projeler geliştirilmesi hedeflenmiştir.

Sevgili eğitmenlerimiz ve öğrencilerimiz başta olmak üzere kitabın bütün eğitim camiamıza katkı sağlaması dileğiyle!

Öğretim Kılavuzu

YAPAY ZEKÂ DERSİ ÖĞRETİM KILAVUZU

Öğrenme, bireyin yaşantılar sonucu davranışlarında meydana gelen oldukça uzun süreli değişimlerdir. Eğitim üçgenine göre eğitmen ve bilgi etkileşimi "öğretim", eğitmen ve öğrenci ilişkileri "pedagoji", öğrenci ve bilgi etkileşimi "öğrenme" sürecini oluşturmaktadır (Amerikan Ulusal Araştırma Konseyi, 2012). İnternet devrimi ve bilgisayar kullanımının yaygınlaşması eğitim alanında da bir "dijital devrim" olarak ortaya çıkmıştır. İçinde bulunduğumuz dönem "bilgi çağı" olarak adlandırılmaktadır. 21. Yüzyıl öğrenenleri için bilişsel, kişisel, kişiler arası alan olmak üzere üç temel yeterlik başlığı belirlemiştir Amerikan Ulusal Araştırma Konseyi (2012). Söz konusu yeterliklerden de hareketle yapay zekanın ele alındığı bu öğretim kitabında da bağlam temelli öğrenme-öğretme ve problem çözmenin bilişsel yaklaşım içerisinde harmanlandığı bir strateji benimsenmiş durumdadır.

Bireyin gerçek hayatta karşılaştığı olgu, olay veya kullanmış olduğu teknolojinin ders içerikleriyle ilişkilendirilmesi bağlam temelli öğrenme-öğretme olarak tanımlanır (MEB, 2012). Öğrenciye aktarılacak yeni bilgiler ile önceki öğrenmeleri arasında kurulan köprü veya bağlantı olarak düşünülebilir. Bu kitapta yapay zekâ çözümleriyle bağlantılı içerik de bireyin gerçek hayatta karşılaştığı, hatta günümüzde güncelliğini koruyan olgular, olay ve teknolojiler çerçevesinde ele alınmıştır. Bu olaylar özellikle yapay zekâ yönünde problem çözme odaklı eylemlerin kullanılması yönünde irdelenmiş ve böylelikle problem çözme yoluyla yapay zekanın (ve ilgili tekniklerin) öğretilmesi hedeflenmiştir.

Bağlam temelli öğrenme adımından sonra konumlandırılan problem çözme, sorunun nedenini belirleyerek çözüm için alternatifleri deneyerek güçlükleri yenme ve istenilen sonuca ulaşma çabasıdır. Problemler, tecrübelerimiz ve öğrenme süreçleriyle elde ettiğimiz bilgi-becerilerinin yapılandırılmasını gerekli kılmaktadır (Jonassen, 2007). Bu kitaptaki yapay zekâ öğretimi de bireylere her yeni yapay zekâ tekniğinin problem çözme odaklı uyarlanmasını önermekte, problem çözümleri yapay zekanın ve ilgili tekniklerin kavranmasını kolaylaştırmaktadır. Bu yolla da esasında bilişsel yaklaşım içerisinde bir süreç de tecrübe edilmektedir.

Bilişsel yaklaşıma göre eğitim, bireyin çeşitli yaşantılar yoluyla zihnindeki şemaları geliştirme sürecidir. Zihinsel şemalar bilgileri düzenleme, yerleştirme ve kullanma biçimlerini içermektedir. Öğrencinin zihnindeki şemaların zengin ve gelişmiş olması alınan bilgilerin daha kolay özümsenmesini sağlamaktadır. Etkili ve verimli bir eğitim için bireylerin zihinsel değişimine katkı sağlayan öğrenme türlerine ağırlık verilmelidir. Öğrencilerin girişimcilik, yaratıcı düşünme, eleştirel düşünme, karmaşık problemleri çözme, etkili iletişim ve takım çalışması gibi becerileri kazanmalarına yönelik olarak tasarlanmalıdır. Benzer şekilde kitap içerisinde sunulan içerik özellikle kodlama çözümleri aracılığıyla bireylerin zihin şemalarını bir arada etkili bir şekilde

kullanarak etkili ve verimli bir bilgi-beceri kazanımı oluşturmalarını sağlamakta, bu yolla yapay zekâ öğretimi kapsamlı bir şekilde tamamlanmaktadır.

Günümüzde bilgi teknolojilerindeki gelişmeler ile bilim arasında doğrudan bir ilişki vardır. Bilgi teknolojileri bilimsel çalışmalar içerisindeki veri düzenlemesi, veri yönetimi-analizi ve elde edilen bulguların yorumlanması gibi aşamalarda aktif bir biçimde kullanılmaktadır (Hacer, 2004). Söz konusu aşamalardan hareketle kitaptaki içerik de algıla, tasarla, harekete geç, yürüt ve karar ver öğrenme döngüleri içerisinde organize edilmiştir. Bu döngüler bağlam temelli öğrenme-öğretme ve problem çözme ile bilişsel yaklaşımı desteklerken, bireylere bilimsel çalışmalarda izleyebilecekleri adımlara aşina olmalarını mümkün kılmaktadır. Söz konusu döngüler ve kitaptaki yapay zekâ uygulamalarıyla bağlantılı Python programlama diline ilişkin genel bilgiler ilerleyen paragraflar altında açıklanmıştır:

ALGILA- TASARLA – HAREKETE GEÇ- YÜRÜT-KARAR VER ÖĞRENME DÖNGÜSÜ

Algıla-Tasarla-Hareket Geç-Yürüt-Karar Ver öğrenme döngüsü adımları, yapay zekanın uygulamalarla etkili öğretilebilmesi için Python programlama dilini de dikkate almak üzere şu şekilde kullanılmalıdır:

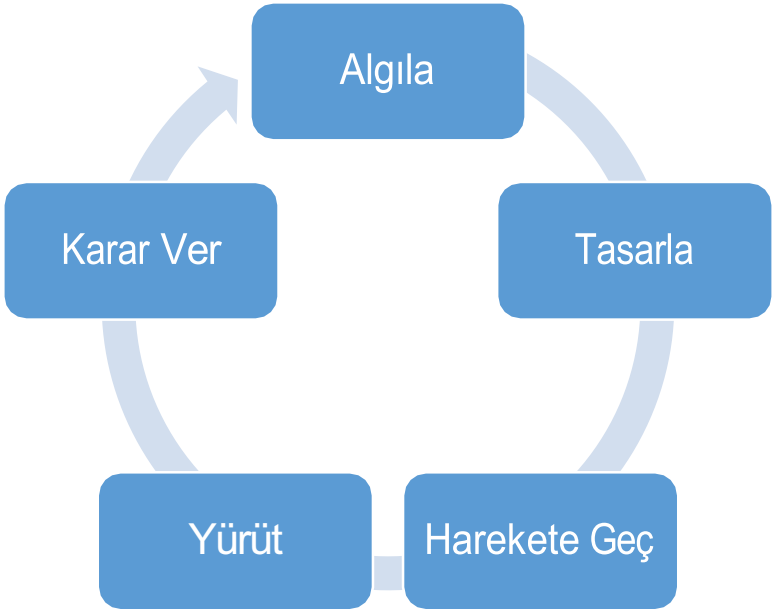
Algıla: Bu bölümde, ilgili hafta ile ilgili eğitmen öğrencilere teorik bilgileri görsel destekler ile aktarır. Anlatılacak konu ile ilgili olasılı avantaj ve dezavantajları detaylı biçimde anlatarak öğrencilerin dersin konusu ile ilgili algısal düşüncelerini arttırır. Eğitmen, Python programlama geçmiş bilgilerini aktive etmek ve Python yapay zekâ kütüphaneleri hakkında öğrencilerin dikkat ve motivasyonlarını sağlamak ile görevlidir. Eğitmenin bir önceki derste yapılan yapay zekâ uygulamalarını mutlaka özetlemesi gerekir.

Tasarla: Bu bölümde eğitmen öğrenciler ilk olarak ilgili hafta ile ilgili belirlenen problemleri anlaması sağlayarak ilgili haftada belirlenen yapay zekâ modeli kullanarak problem çözmesi sağlanır. Bunun için Python programlama editöründe yapay zekâ veri setini öğretme ve hazırlama işlemlerini yapar. Öğrenciler, eğitmenin göstermiş olduğu kodları Python programlama editöründe uygulamaya gerçekleştirir. Bu bölüm dahil, ilerleyen uygulama odaklı bölümlerde dersteki kodlar ve veri setleri için Github üzerinde https://github.com/deneyapyz/ortaokul/ platformu tanımlanmıştır.

Haraket Geç: Bu bölümde öğrenciler Python editöründe kod yazmak için aktif rol alırlar. Eğitmen uygulayıcı pozisyonundadır. Ayrıca öğrencilerin kod yazarken yaptıkları hatalarda destek olacaktır. Fakat eğitmenin sağladığı destek gereğinden fazla da olmamalıdır. Eğitmen öğrencilerden eğitim ve test verilerini ayırmasını ve model kurmada başarılı olmasını bekler.

Yürüt: Bu bölümde eğitmen Python kodlama ekranında program kod satırlarını adım adım yazar. Öğrenci ise eğitmenin kod yazdığı her bir adımdan sonra kodu uygulayarak ilgili haftaya ait uygulamayı gerçekleştirir. Burada, öğrenciler Python editöründe kod yazmak için aktif rol alırlar. Eğitmen uygulayıcı pozisyonundadır. Eğitmen öğrencilerden modeli eğitme ve tahminlenmesini doğru biçimde kodlamasını bekler. Öğrencilerin takıldıkları noktalarda destek olacaktır.

Karar Ver: Öğrenci Python editöründe gerçekleştirmiş olduğu yapay zekâ uygulamasındaki başarısını test eder. Elde edilen test sonuçlarını hata matrisi üzerinde gözlemleyerek yapay zekâ modeli başarısını test eder. Github platformundaki ders klasörü ile etkileşim devam eder.



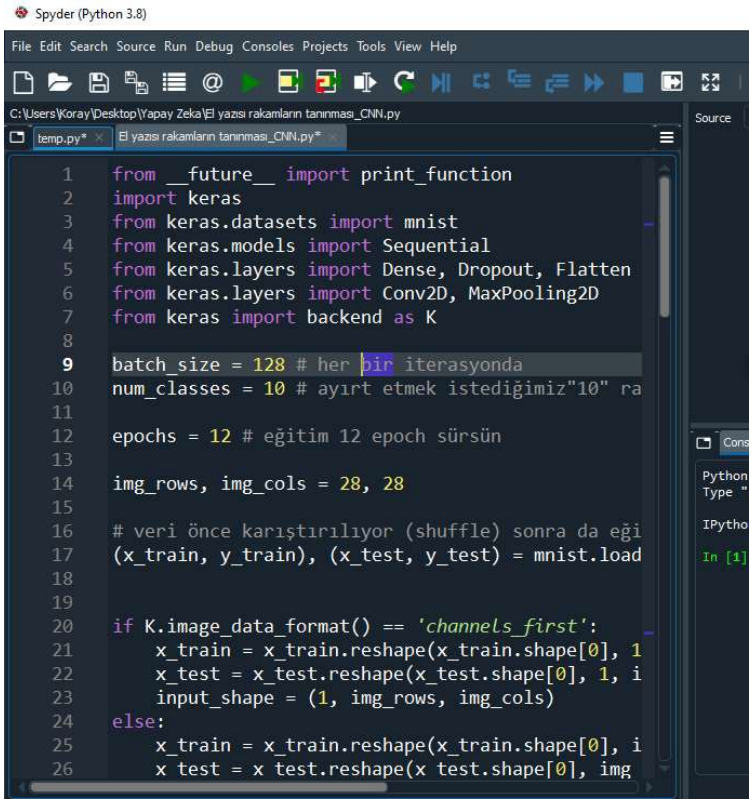
Şekil 1. Öğrenme Döngüsü

PYTHON PROGRAMLAMA ORTAMI

Python

Günümüzde teknolojinin hızka gelişmesi ile bilgisayar progralama dillerine olan ilgide gün geçtikçe artmaktadır. C++, Visual Studio.Net, Python gibi birçok yaygın biçimde programlama dili kullanılmaktadır. Özellikle açık kaynak kodlu olan ve sürekli olarak kütüphaneleri geliştirilen Python programlama dili bu diller içerisnde öne çıkan programlama dilidir. Python programlama dili basit bir kod yapısına sahip olduğu için öğrenilmesi oldukça kolaydır. Python kolay öğrenilebilirliği, ücretsiz ve açık kaynak kodlu olmasından dolayı sürekli geliştirilebilmesi sayesinde kodlamaya yeni başlayanların ilk tercihi olmaktadır. Python, 2020 IEEE araştırmasına göre dünya çapında en çok kullanılan ve tercih edilen programlama dillerinden birisidir (Cass, 2020). Yapay zekâ denince akla ilk olarak Python dili gelmektedir. Bu durum da Python'ın dünya çapında büyük bir kitlesinin olmasına neden olmaktadır. Ancak yapay zekâ uygulamalarında Matlab, C++, Lush gibi standart dillerde kullanılır. Python programlama dilinde yapay zekâ işlemede genellikle Keras, Numpy, Pytorch, Matplotlib, Scipy, Scikit-Learn kütüphaneleri kullanılmaktadır. Python programlama dilinin diğer programlama dillerinden farkı, çok yüksek seviyeli olması dolayısıyla az sayıda komutla çok sayıda işlemin pratik şekilde gerçekleştirilebilmesi, nesne yönelimli programlama yaklaşımının hızlı bir şekilde hayata geçirilebilmesi ve her geçen zaman genişleyen, aktif bir kütüphane havuzuna sahip olmasıdır. Python'da IDLE, Spyder ve Pycharm gibi farklı kod yazma editörleri kullanabilmektedir. Sahip olduğu yetenekler dolayısıyla Python programlama dili üzerinden internet / web platformları

geliştirme, bilimsel ve sayısal hesaplama süreçleri işletme, yapay zekâ sistemleri oluşturma, robotik çözümlerin kod altyapısını destekleme gibi birçok farklı alanda uygulamalar gerçekleştirilmektedir.



Şekil 2. Python Spyder kodlama ortamı.

Kitap içeriği hem yüz yüze hem de uzaktan eğitime uyumlu bir yapıda inşa edilmiştir. Özellikle uzaktan eğitim süreçlerinde, içerikte sunulan açıklamaların ve yönlendirici kutucukların dikkate alınması öğretim-öğrenim süreçlerinin kalitesini artırabilecektir. Kitapta içerik akışı Python ile yapay zekâ, yapay zekâ matematiği ve bulanık mantık, olasılık çözümleri için bayes öğrenme, karar ağaçları makine öğrenme algoritması, yapay sinir ağları, etmen tabanlı modelleme, zeki optimizasyon ve derin öğrenme olmak üzere 8 haftalık program altında kullanıma sunulmuştur. Öğrenciler 8 haftalık program sonucunda şu kazanımları elde etmektedir:

- Algoritmik yapıları öğrenir ve basit bir yapay zekâ oluşturur.
- Python ile bulanık mantık tabanlı matematiksel işlemleri gerçekleştiren yapay zekâ tabanlı çözüm üretir.
- Makine ve Bayes öğrenme ile regresyon, sınıflandırma ve kümeleme yöntemlerini uygun olanı modeller.
- Karar ağaçları algoritmasının kullanarak örnek uygulama yapar.
- Yapay sinir ağları modeli ile yüksek bir başarım oranı ile model oluşturur.
- Etmen tabanlı modeller davranışlarını kullanarak mimari tasarlar ve problemi çözer.
- Zeki optimizasyon teknikleri kullanarak koloni optimizasyon davranışını uygular.
- Yapay Zekâ alanı açısından derin öğrenme algoritmalarını kullanarak örnek uygulamalar yapar.

Kitap içeriği Python programlama dili içerdiği için yapay zekâ odaklı konuların anlaşılması konusunda Python kodlama bilgi ve becerisi de gerektirmektedir. Kitap üzerinden öğretim ve öğrenim süreçlerine geçilmeden hemen önce Python ile ilgili temel eğitimler alınabileceği gibi, https://owncloud.tubitak.gov.tr/index.php/s/6wbNcqcaJh8h1OZ adresi altında sunulan 17 videoluk anlatım serisi ile Python kodlama temelleri öğrenilebilmektedir. Öğrencilerin özellikle ilgili video serisinden destek alarak, 1 haftalık bir Python ön-öğrenim süreci geçirmeleri kitapta anlatılanların daha iyi anlaşılması adına önemlidir. Yine videolar altında anlatılan örnek kodlar https://github.com/deneyapyz/pythonvideolari Web adresinde sunulmuş durumdadır.

Kitaptaki öğretim sürecinin etkin olması adına çeşitli Web unsurlarından ve özel içerik kutucuklarından faydalanılmıştır. Söz konusu bileşenlerin genel işlevleri kısaca şöyledir:

- **GitHub:** Güncel yazılım projelerinin ve geliştirici profillerinin paylaşıldığı GitHub platformunun öğrenciler tarafından önerilmesi önemli olduğu için kitap içeriğinde anlatılan yapay zekâ konularıyla bağlantılı kodlar ve veri setleri içerikte aktarılan GitHub linkleri altında kullanıma açılmıştır.
- **QR (Kare) Kodlar:** Öğrencilerin ders ve öğretim süreçlerine ilgilerini aktif tutmak ve yapay zekâ çerçevesinde daha etkin ve güncel fikirler oluşturabilmek adına çeşitli video linklerine bağlantılar sunulmuştur. Öğrenciler (ve hatta eğitmenler) ders süreçlerinde ilgili QR (kare) kodları okutup videolara erişebilecek, ders sırasında videoları tartışabilecek ya da ders harici zamanlarda videoları izleyerek güncel konulara dair kazanımlar edinebilecektir.
- **Eğitmene Not:** Eğitmene Not kutucukları, kitap içeriğinin öğrencilere aktarımı aşamasında eğitmenlere tavsiyeler ya da bazı kritik işlem adımlarını aktarabilmek adına kullanılmıştır.
- **Düşün, tartış…:** Düşün, tartış kutucukları ders sırasında yapay zekâ ile ilgili güncel konulardan hareketle sınıf içi ya da bireysel fikirlerin oluşturulup tartışılmasını sağlamak için kullanıma sunulmuştur. Bu kutucuklar altındaki sorular (tartışma konuları) özellikle öğrencilerin entelektüel bilgi düzeylerini güncel konularla harmanlamayı hedeflemektedir.
- **Biliyor musunuz?** Biliyor musunuz kutucukları yapay zekâ ile ilgili güncel, önemli ve dikkate çekici konuları / gelişmeleri öğrenciler ile kısa notlar üzerinden paylaşmayı hedeflemiştir.
- **Dünyadan Haberler:** Dünyadan Haberler kutucukları, ders haftaları içerisindeki yoğun kodlama süreçlerini esnekleştirmek ve güncel yapay zekâ gelişmelerini öğrencilere kaynaklar üzerinden aktararak bilgi birikimi desteklemek adına sunulmuştur.

Proje Yarışması: Kitap ile sağlanan yapay zekâ eğitimini etkin bir proje içerisinde harmanlamak ve öğrencilerin elde ettikleri bilgi-becerilerini nihai bir çalışma altında, rekabetçi bir yönde işe koşmasını sağlamak için proje yarışması oluşturulmuştur. Detayları kitabın sonunda sunulmuş olan proje yarışması, iklim değişikliği (krizi) özelinde hedef bir veri seti (problem) için öğrencilerin özgür bir şekilde yapay zekâ çözümü üretmelerini ve ürettikleri çözümün taşıdığı niteliklere göre puanlanmasını içeren prosedürleri kapsamaktadır. Yarışma süreci 7. hafta sonundan 8. hafta dersi 2 gün öncesine kadar sürmektedir.

Buraya kadarki açıklamalardan da hareketle, haftalık öğretim sürecinin şöyle bir akış içerisinde gerçekleştirilmesi önerilmektedir: Eğitmen öğrencilere ilgili haftaya ait konuyu teorik olarak detaylı biçimde aktarır. Sonrasında, öğrencilere ilgili haftanın veri seti, flash disk ortamında ya da çevrim içi indirme linkleri yoluyla verilir. Eğitmen tarafından öğrenciler ile birlikte Python programlama dilinde ilgili haftaya ait örnek uygulamayı kod olarak yazar ve yürütür. Eğitmen ve öğrenci birlikte uygulama kodlarından elde edilen sonucu değerlendirir. Bu noktada içerikte sunulan eğitmene not kutucukları ve diğer destekleyici kutucuklar da dikkate alınarak, öğretim-öğrenim süreçlerinin etkinliği artırılır.

Kaynakça

National Research Council. (2012). Education for life and work: Developing transferable knowledge and skills in the 21st century. National Academies Press.

Meb 2011, Bağlam Temelli Öğrenme-Öğretme Yaklaşımı. Available from: https://www.researchgate.net/publication/334959478_BAGLAM_TEMELLI_OGRENME-OGRETME_YAKLASIMI [accessed Jun 14 2021].

Jonassen, D. (2011). Designing for problem solving. In R. Reiser & J. Dempsey (eds.), *Trends and Issues in Instructional Design and Technology*, (pp. 64–74). Boston, MA: Pearson Education.

Hacer, T. O. R., & Erden, O. (2004). Ilkögretim ögrencilerinin bilgi teknolojilerinden yararlanma düzeyleri üzerine bir arastirma. TOJET: The Turkish Online Journal of Educational Technology, 3(1).

Cass, S. (2020). The top programming languages: Our latest rankings put Python on top-again-[Careers]. IEEE Spectrum, 57(8), 22-22.

1. Hafta: Python ile Yapay Zekâ

Ön Bilgi:

- Temel bilgisayar programlama ve algoritma bilgisi verilecektir.
- Python anlatım videoları 1. Python Dili Özellikleri, 2. Python Yukleme Kodlama Ort., 3. Python Kodlama İçin Spyder, 4. Veri Tipleri, 5. Veri Tipleri Dönüşümleri, 6. Atama Operatörleri ve Aritmetiksel Operatörler ve ayrıca 7. Karşılaştırma Op., Mantıksal Operatörler. 8. If Yapısı, 9. Giriş-Çıkış Komutları, 10.-11. Döngüler ve 12. Veri Yapıları-Veri Modelleri öğrenciler tarafından hafta öncesinde izlenmiş olmalıdır.

Haftanın Kazanımları:

- Öğrenciler "yapay zekâ" kavramını temel düzeyde açıklar.
- Öğrenciler algoritma kavramını açıklar.
- Öğrenciler basit bir yapay zekâ örneği oluşturabilir.
- Öğrenciler Python ile yapay zekâ temel kütüphanesi ile giriş seviyesinde uygulamalar yapar.
- Öğrenciler Github platformu ile yapay zekâ örnekleri üzerinden karar vermeyi öğrenir.

Haftanın Amacı:

Bu haftanın amacı, "yapay zekâ" kavramının tüm öğrenciler tarafından doğru bir şekilde anlaşılmasını sağlamaktır. Haftanın bir diğer amacı, Python ile örnek kodlar yazarak yapay zekâ uygulamaları ile öğrencilerin ilgisini çekerek etkinliklere yönelik ilgilerini artırmaktır. Ayrıca, öğrencilerin problem algılaması ve bu probleme ait yapay zekâ modelleme kavramının ne olduğunu tanımlaması ve basit bir veri organizasyonu oluşturabilmesi de hedeflenmektedir. Ders süresince kullanılacak Python editör ve temel yapay zekâ kütüphanelerini tanıtmak ve öğrencilerin bu kütüphaneleri kullanarak temel düzeyde yapay zekâ uygulamaları yapmaları hedeflenmektedir.

Kullanılacak Malzemeler:

Bilgisayar, kâğıt, kalem, Python kod editörü

Haftanın İşlenişi:

Algıla: Yapay zekâ kavramı üzerine tartışma ve problem çözüm şekilleri algılayabilir.

Tasarla: Yapay zekâda veri organize etme işlemlerini temel düzeyde gerçekleştirebilir.

Harekete Geç: Python ile eğitim ve test verilerini ayırma, model kurma kurma gibi süreçleri gerçekleştirebilir.

Yürüt: Eğitmenlerin öğrencilerin Python ile model eğitme ve tahminleme işlemlerini gerçekleştirebilmeleri için uygulamalara birebir ve etkili rehberlik eder. Böylelikle öğrenciler de yapay zekâ model eğitme ve tahminleme çözümlerini uygulayabilir.

Karar Ver: Öğrenciler alternatif yapay zekâ modelini düzenleyerek / kodlayarak çözüm üretebilir.

1. ALGILA

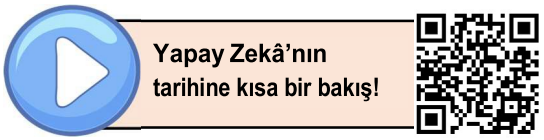
1.1. Yapay Zekâ Kavramı Üzerine Tartışma

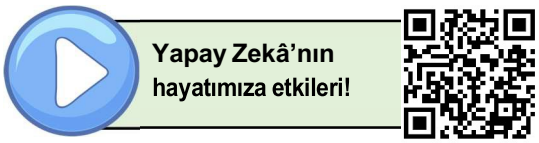
Eğitmen, öğrencilerin "yapay zekâ" kavramı üzerinde tartışmalarını sağlar.

- Yapay zekâ nedir?
- Yapay zekâ bileşenleri nelerdir?
- Yapay zekâda problem çözüm şekilleri nelerdir?
- Yapay zekâ ile nerelerde karşılaşabiliriz?
- Robotlar yapay zekâ ile dünyayı ele geçirebilirler mi?
- Yapay zekâ insan beynini nasıl taklit ediyor?
- Yapay zekâ hayatımızı nasıl şekillendirecek?
- Python nedir?
- Python programlama dili ile yapay zekâ nasıl öğrenilir?
- Python için hangi temel yapay zekâ kütüphaneleri kullanılır?
- Program/modelleme nedir?

Sorularını yönelterek tartışmayı yönetir. Cevapları öğrencilerin bulması sağlanır. Eğitmen, yapay zekâ ve bileşenleri, program ve modelleme kavramları için ilerleyen satırlarda verilen tanımları dikkate alarak öğrencileri yönlendirebilir veya öğrenciler kavramı yanlış tanımladıklarında tanımlamaları doğrudan kendisi yapabilir.

Yapay zekâ, insan zekâsını modelleyebilmek adına insan gibi akıl yürütme, anlam çıkartma, genelleme yapabilme, geçmiş deneyimleri ile öğrenebilme gibi yetkinlikleri bir bilgisayara ya da makineye kazandırabilmektedir. *Oxford İngilizce Sözlük*'te ise yapay zekâ, görsel algılama, konuşma tanıma, karar verme ve diller arasında çeviri gibi normalde insan zekâsını gerektiren görevleri yerine getirebilen bilgisayar sistemleri olarak tanımlanmaktadır. Her iki tanımda da vurgulanan nokta, yapay zekânın bir görevi gerçekleştirirken insan beyni gibi öğrenme fonksiyonunu gerçekleştiren bilgisayar sistemleri olmasıdır. Yapay zekâ, insanlara göre daha akıllı çalışma yeteneğine sahip olup daha hızlı akıl yürütmekte ve daha doğru karar vermektedir (Yılmaz, 2021).





Yapay zekâ Şekil 1.1'de gösterildiği gibi veri seti, öğrenme algoritmaları ve karar verme süreci olmak üzere üç ana bileşenden oluşur. Yapay zekâda metin, görüntü verileri, zaman, uzunluk ölçümleri, video kayıtlarının düzenlenmesi ile veri seti oluşmaktadır. Öğrenme algoritmaları yapay sinir ağları, makine öğrenmesi, derin öğrenme gibi birçok alt daldan oluşmaktadır. Yapay zekâ

öğrenme algoritmaları karmaşık bir yapıya sahip gibi gözükse de temelde algoritma ve programlamadan oluşmaktadır.

Şekil 1.1. Yapay zekâ bileşenleri

Algoritma: Bilgisayarlarda bir problemin çözümünde izlenecek yol genel tanımıyla algoritma olarak adlandırılır. Çocuklar günlük hayatta kullandıkları tabletlerden, oynadıkları oyunlara, yaşamlarının her alanında algoritmaları kullanmaktadır. Algoritma mantığı veya bir problemi adım adım çözebilme yeteneği özellikle fen ve matematik gibi sayısal derslerde oldukça önemlidir. Örneğin matematik dersinde bölme, çarpma ve çıkarma gibi işlemleri yaparken algoritmalardan faydalanılır. Çocukların, problemleri bölümlere ayırma ve çözüme ulaşabilmek için yapmış oldukları işlem basamakları algoritmik düşünceye örnek olarak verilebilir.

Program: Yapay zekâ uygulamaları için algoritma adımları bilgisayar tarafından gerçekleştirilen program kodlarının bütünüdür. Programlar, yapay zekâ uygulamalarının bilgisayar tarafından yapılması için gereken adımların bir programlama dili aracılığıyla aktarılmasıdır. Öğrenciler de bu derste yapay zekâ uygulamaları geliştirmek için Python programlama dili kullanacaktır.

Düşün, tartış…

İnsanlar teknolojiye neden ihtiyaç duymaktadır? Yapay Zekâ'nın buradaki rolü nedir; neden bu kadar önemli ve popüler olmuştur?

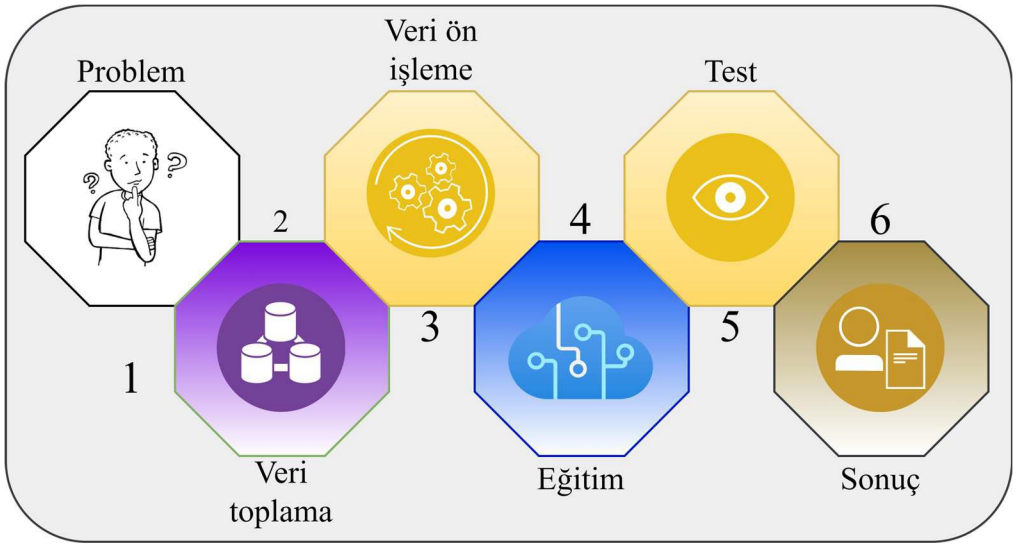
1.2. Yapay Zekâda Problem Modelleme

Yapay zekâda problemin modellemesinde kullanılan birçok otomatik karar verme mekanizması vardır. Yapay zekâ modelleri ile veri girişi sağlandığında uzman bir kişinin vereceği kararı "eğitilmiş" matematiksel algoritmalar yapar. Ayrıca yapay zekâ modelleri karar sürecinin yorumlanmasına yardımcı olurlar. Yapay zekâ modelleri, sistemin en doğru karar vermesi veya maliyeti en aza indirmesi için büyük miktarda veriyi işlemektedir. Farklı ortamlardan gelen verileri analiz eden yapay zekâ modeli tüm verileri gözden geçirerek uzmanlardan oluşan bir ekibin yapacağı belirli bir karar sürecini tek başına gerçekleştirir. Şekil 1.2'de gösterildiği gibi yapay zekâ modellemesi veri akışı şöyledir:

- **Problem:** Öğrenciler yapay zekâ çözümleri için problemin sınıflandırma, regresyon veya kümeleme türünü belirler. Örneğin kedi ve köpek verilerinden oluşan veri setinin

sınıflandırma problemi olduğunu, yaya sayısına göre trafik ışıklarının süresinin belirlenmesinin bir regresyon problemi olduğunu analiz eder.

- **Veri Toplama:** Öğrenciler problemin çözümü ile ilgili açık erişimli internet sitelerinde yer alan (Kaggle, Github vs.) veya kendi toplamış oldukları verileri bir araya getirir.
- **Veri Önişleme:** Bu aşamada, toplanan veriler üzerinde eksik verileri tamamlama, anlamsız verileri çıkarma gibi veri önişleme aşamaları gerçekleştirilerek yapay zekâda oluşabilecek problemlerin önlenmesi sağlanır.
- **Eğitim:** Veri önişleme sonrasında anlamlandırılan verilerden eğitim için ayrılan verileri yapay zekâ modelleri ile eğitir.
- **Test:** Eğitilen modellerin doğruluğu test verileri ile değerlendirilerek anlamlı sonuçlar elde edilmeye çalışılır.
- **Sonuç:** Test edilen veriler üzerinde en doğru sonuç veren modeli seçer.



Şekil 1.2. Yapay zekâ problem modelleme

Biliyor musunuz?

Yapay Zekâ'nın en etkili kullanıldığı alanlardan biri de sağlıktır. Sağlık alanında hastalıkların erken teşhisinde, tedavi planlamasında ve hatta ilaç keşfinde Yapay Zekâ yaygın bir şekilde kullanılmaktadır. Yapay Zekâ bu alanda doktorlarla yarışmaktadır!

1.3. Python Programlama Dilinde Kullanılan Temel Yapay Zekâ Kütüphaneleri

Python, kolay öğrenebilen, nesne tabanlı uygulamaları ile oldukça sık tercih edilmektedir. Python programlama dilinde yapay zekâ işlemede genellikle Numpy, Matplotlib, Scipy, Scikit-Learn, TensorFlow, Keras, Pytorch kütüphaneleri kullanılmaktadır. Yapay zekâ modellemesi için yapay zekâ kütüphanelerin kısa açıklaması şu şekildedir:

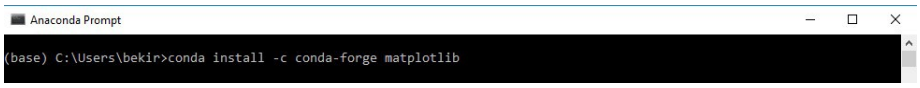
Numpy: Hesaplama işlemleri, diziler ve diziler üzerinde hızlı işlemler yapılabilmesi için kullanılan önemli kütüphanelerden birisidir (Şekil 1.3).

Şekil 1.3. Anaconda Spyder için Numpy kütüphanesinin kurulumu

Matplotlib: Görüntüleri yüksek kalitede gösterebilmek için kullanılan bir çizim kütüphanesidir (Şekil 1.4).

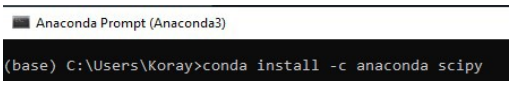
Şekil 1.4. Matplotlib kütüphanesi ile ilgili örnek grafikler

Anaconda Spyder'da Matplotlib kütüphanesini yüklemek için anaconda prompt ekranında Şekil 1.5'de görüldüğü gibi "conda install -c conda-forge matplotlib" komut satırının yazılması gerekmektedir.



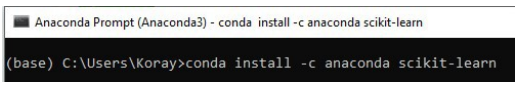
Şekil 1.5. Anaconda Spyder için Matplotlib kütüphanesinin kurulumu

Scipy: Veri analizinde kümeleme, regresyon (tahmin), veri işleme gibi işlemleri gerçekleştirebilen çok yönlü bir kütüphanedir (Şekil 1.6).



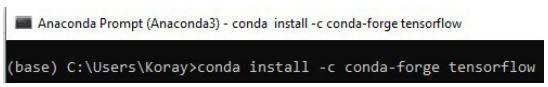
Şekil 1.6. Anaconda Spyder için Scipy kütüphanesinin kurulumu

Scikit-Learn: Makine öğrenme modelleri oluşturmak için kullanılan bir kütüphanedir. Regresyon (tahminleme), kümeleme ve sınıflandırma için kullanılan pek çok öğrenme algoritmasına sahiptir (Şekil 1.7).



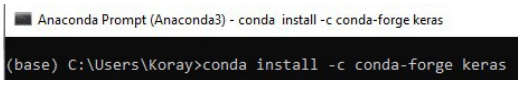
Şekil 1.7. Anaconda Spyder için Scikit-Learn kütüphanesinin kurulumu

TensorFlow: Google tarafından geliştirilen açık kaynaklı kodlu bir derin öğrenme kütüphanesidir (Şekil 1.8).



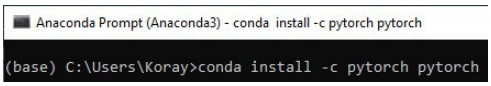
Şekil 1.8. Anaconda Spyder için TensorFlow kütüphanesinin kurulumu

Keras: Derin sinir ağları ile hızlı eğitim yapabilmek için tasarlanan açık kaynak kodlu bir sinir ağı kütüphanesidir (Şekil 1.9).



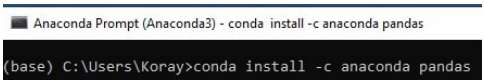
Şekil 1.9. Anaconda Spyder için Keras kütüphanesinin kurulumu

Pytorch: Derin öğrenme modellerinde esneklik ve hız ile geliştiricilerin işlerini oldukça kolaylaştıran bir kütüphanedir (Şekil 1.10).



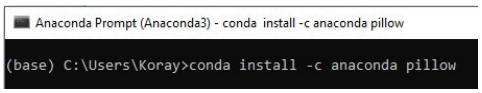
Şekil 1.10. Anaconda Spyder için Pytorch kütüphanesinin kurulumu

Pandas: Tablosal veri işlemesi ve analizi için kullanılan Python temel kütüphanesidir (Şekil 1.11).



Şekil 1.11. Anaconda Spyder için Pandas kütüphanesinin kurulumu

Pillow: Görüntü işleme için kullanılan Python temel kütüphanesidir (Şekil 1.12).



Şekil 1.12. Anaconda Spyder için Pillow kütüphanesinin kurulumu

Biliyor musunuz?

Türkiye'nin en önemli bilim insanlarından birisi de dünyaca ünlü matematikçi, birçok matematiksel kavramın mucidi **Ord. Prof. Dr. Cahit Arf**'tır.

Cahit Arf, dünyanın daha tanıştığı Yapay Zekâ teknolojisini düşünerek, '*Makine Düşünebilir mi ve Nasıl Düşünebilir?*' adlı çalışmasıyla Yapay Zekâ'nın gücünü 20. Yüzyıl ortalarında tartışmaya açmıştır. Söz konusu çalışma Atatürk Üniversitesi Üniversite Çalışmalarını Muhite Yayma ve Halk Eğitimi Yayınları Konferanslar Serisi altında yayınlanmıştır. Bu bakımdan Prof. Arf, Yapay Zekâ ve düşünen makine konusuna yönelik gösterdiği öngörüler bakımından dünyadaki sayılı bilim insanları arasındaki yerini almıştır.

Ülkemizin önemli değerlerinden olan Prof. Arf, günlük hayatın koşturması arasında dikkatinizden kaçmış olabilir. Hemen 10 TL'lik kâğıt banknotun arkasına bakabilirsiniz!

Prof. Arf'ın bilim dünyasına kattığı önemli kavramlar hakkında bilgi sahibi olmak için şu anahtar kelimeleri Web'le aratabilirsiniz: Arf Değişmezleri, Arf Halkaları, Arf Kapanışı

1.4. Python ile Yapay Zekâ Veri İşleme

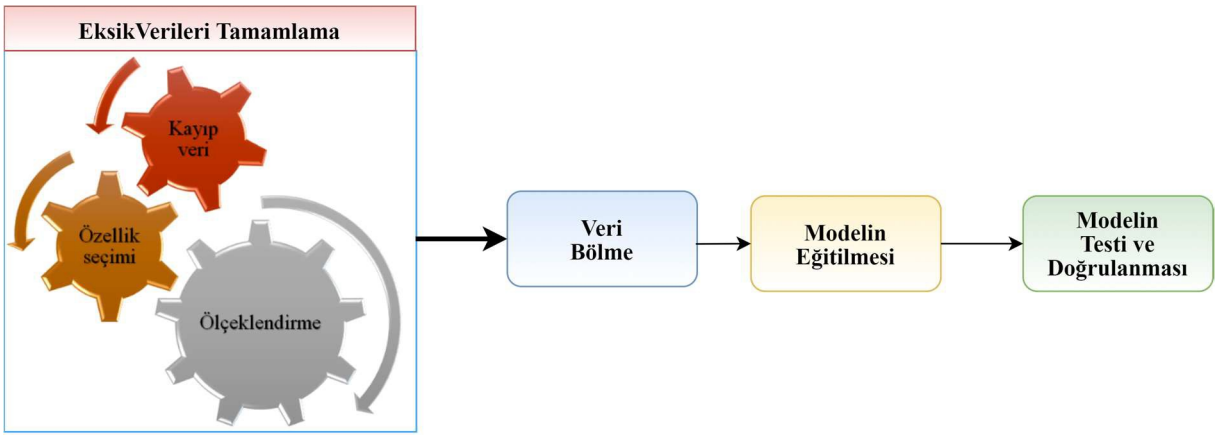
Yapay zekâda veri işleme hem makine öğrenmesi hem de derin öğrenme modellerinden doğru bir biçimde yararlanmak için kullanılan bir tekniktir. Makine öğrenimi yapay zekânın bir alt kümesidir, bilgisayarların verilerden anlamlı sonuçlar elde edilmesini sağlayan önemli tekniklerden birisidir. Derin öğrenme ise, beyindeki sinir ağlarını örnek alarak çalışan karmaşık sorunların çözülmesini sağlayan makine öğreniminin alt kümesidir. Yapay zekâda toplanan veriler üzerinde; eksiklik, gürültü (yanlış veri) ve tutarsızlık gibi farklı nedenlerden dolayı veri işleme problemleri yaşanabilir. Şekil 1.13'te gösterildiği gibi yapay zekâ veri işleme aşamaları şu şekilde sıralanmaktadır:

Eksik Verileri Tamamlama: Yapay zekâ veri işleme sürecinde ilk aşamada toplanan verilerdeki eksik değerler bulunur. Eksik verileri tamamlamak için öznitelik oluşturma, sınıflandırma, ölçeklendirme-normalize etme ve uç verileri tespit etme yöntemleri kullanılır.

Veri Bölme: Yapay zekâda veriler eğitim ve test olmak üzere ikiye ayrılır. Eğitim verisi, modelin eğitildiği verileri, test verisi ise modelin eğitilmeyen veriler üzerindeki sonuçlarını görmek için kullanılır.

Modelin Eğitilmesi: Yapay zekânın doğru bir tahmin yapabilmesi için temizlenmiş verilerin eğitilmesi gerekir.

Modelin Testi ve Doğrulanması: Test verileri ile eğitilen model doğrulanarak yapay zekâ modeli değerlendirilir.



Şekil 1.13. Yapay Zekâ ile Veri Analizi

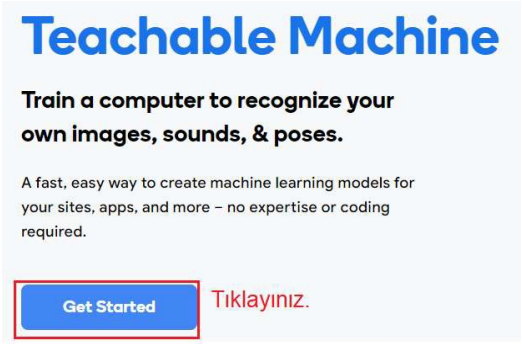
Eğitmene Not

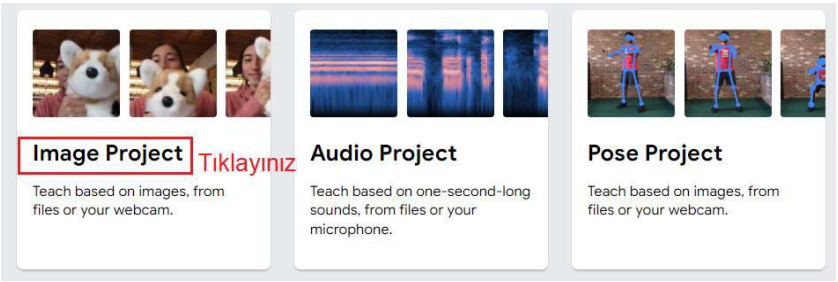
Eğitmen, kitap başlangıcında ifade edilen; iklim değişikliği odaklı proje yarışması konusundaöğrencileri bilgilendirir. Bu noktada, kitap sonunda yer alan açıklamaları dikkate almak sureliyle genel bir bilgilendirme yaparak öğrencileri proje yarışması konusunda ilk haftadan itibaren motive eder. Genel motivasyon iklim değişikliği (krizi) konusunun ciddiyetini ve yapay zekanın bu yöndeki bilimsel araştırmalar konusundaki çözüm potansiyelini irdelemek üzerine olmalıdır (Proje yarışmasının amacı ve genel yapısı hakkında bilgiler için kitap başlangıcındaki öğretim kılavuzu tekrar incelenmelidir).

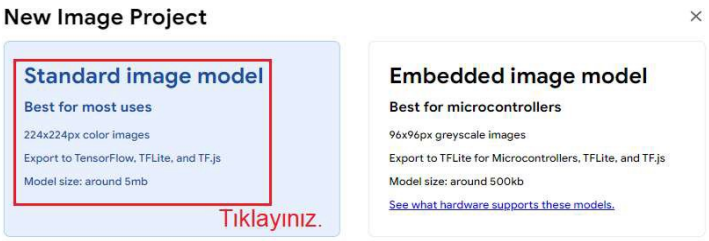
1.5. Eğitilebilir Yapay Zekâ Platformu ile Taş Kâğıt Makas Oyununun Modellenmesi

Öğrenciler aşağıda verilen linki tıklayabilir.

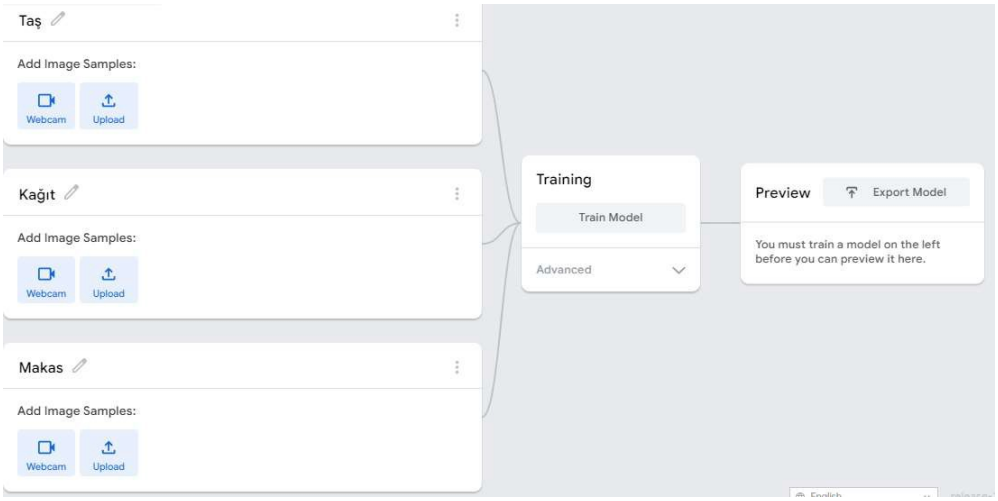
https://teachablemachine.withgoogle.com/



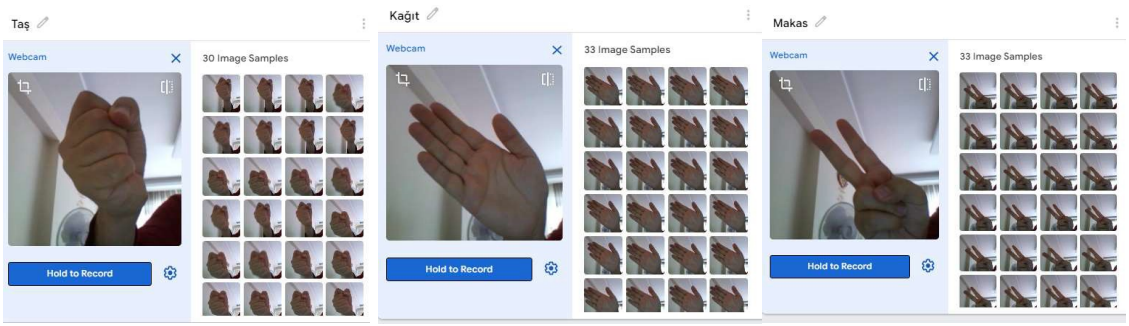




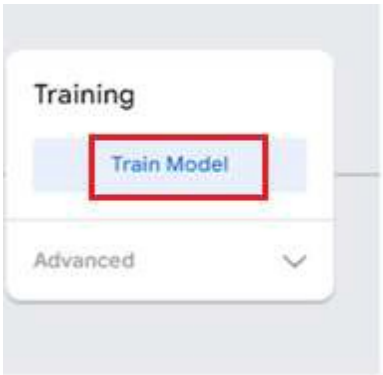
Öğrenciler class isimlerini “Taş”, “Kâğıt” ve “Makas” olarak isimlendirir.



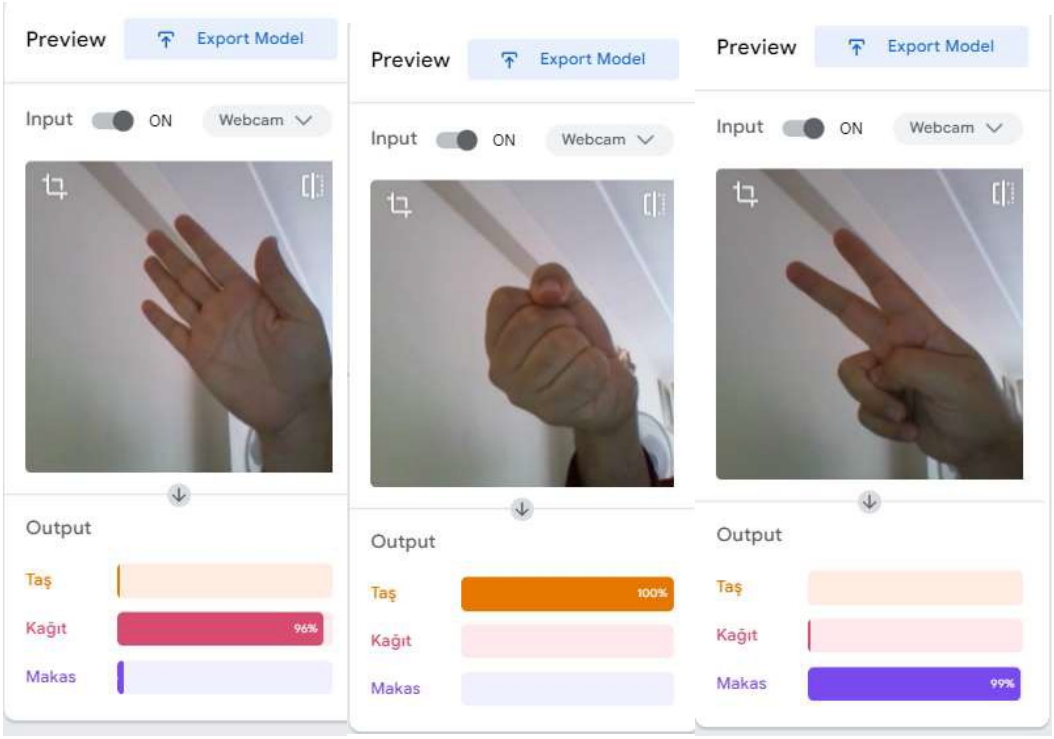
Öğrenciler “hold to record” butonuna basarak taş, kâğıt ve makas oyunu için görüntü veri seti oluşturur.



Öğrenciler “train model” butona basarak görüntü veri setinin eğitim işlemini gerçekleştirir.



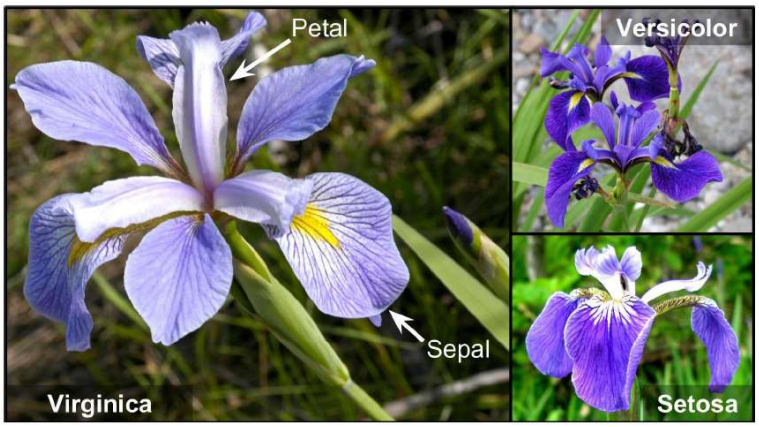
Öğrenciler modeli eğittikten sonra webcam aracılığıyla modelin doğruluğunu test eder.



2. TASARLA

2.1. Veri Setini Öğreniyorum

Sınıftaki öğrencilere Şekil 1.14'te görüldüğü gibi iris bir çiçeğin üç türüne *(setosa, versicolor, virginica)* ait fotoğraf gösterilir. Daha sonra üç türe ait 50'şer tane, toplamda 150 tane olmak üzere üst ve alt çiçek yaprakları ölçülmüş veri seti verilir. Bu ölçümden dört nitelikli *"alt yaprak uzunluğu" cm, "alt yaprak genişliği" cm, üst yaprak genişliği" cm, "üst yaprak uzunluğu" cm* ve 150 elemanlı bir veri seti gösterilir.



Şekil 1.14. İris çiçeği ve türleri (O'Reilly, 2021)

Her bir öğrenci Çizelge 1.1'de verilen iris çiçeğine ait yaprak ölçülerini ve türünü inceler ve karşılaştırır. Öğrenci veri seti tablosundaki alt ve üst yaprak uzunluğunu-genişliğini giriş parametresi olarak isimlendirir. Ölçümlerin değerlerine göre iris çiçeği yaprak türlerini ise çıkış

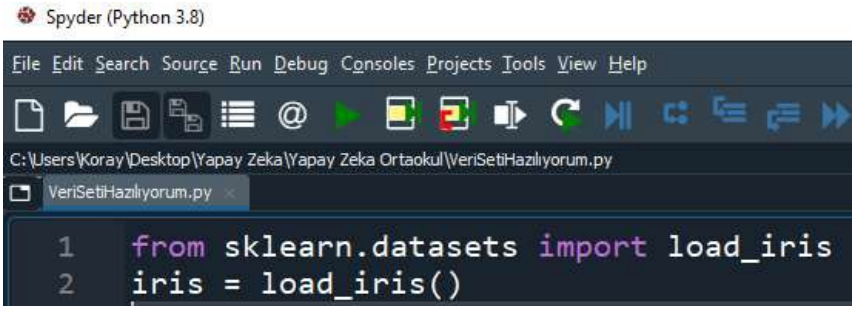
parametresi olarak isimlendirir. Öğrenci veri setindeki ondalıklı sayılarda virgül yerine noktanın kullanıldığını öğrenir.

Çizelge 1.1. İris çiçeği türlerinin yaprak ölçüm değerleri

Örnek Numara	Alt yaprak uzunluğu (cm)	Alt yaprak genişliği (cm)	Üst yaprak uzunluğu (cm)	Üst yaprak genişliği (cm)	Tür
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...	...	...	...	...	...
51	7.0	3.2	4.7	1.4	Versicolor
52	6.4	3.2	4.5	1.5	Versicolor
...	...	...	...	...	...
101	6.3	3.3	6.0	2.5	Virginica
102	5.8	2.7	5.1	1.9	Virginica
...	...	...	...	...	...
150	5.9	3.0	5.1	1.8	Virginica

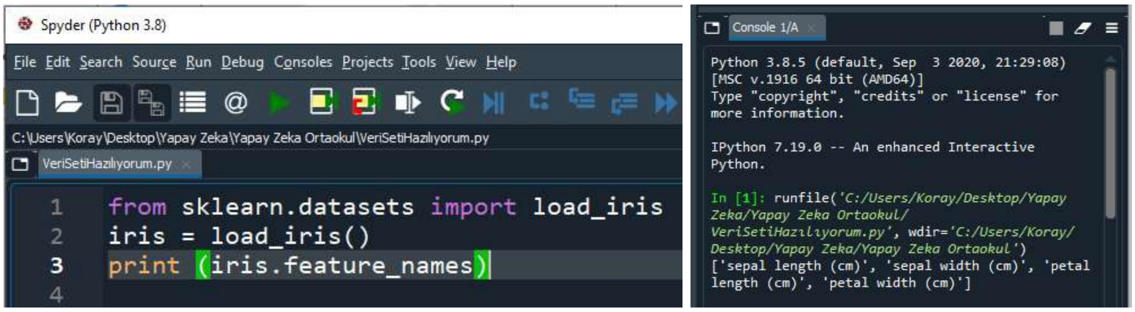
2.2. Python ile Veri Seti Hazırlıyorum

Öğrenci Python scikit-learn kütüphanesinin içinde hazır olarak yer alan iris veri setini kullanarak basit bir sınıflandırma yapar. Tasarım aşamasında iris çiçeğinin alt ve üst yaprak uzunluk ve genişlik verilerini kullanarak çiçeğin türünü sınıflandırmaya çalışır. Şekil 1.15'te gösterilen gerekli kütüphane ve veri setini indirir.



Şekil 1.15. Gerekli kütüphaneleri ve veriyi alma

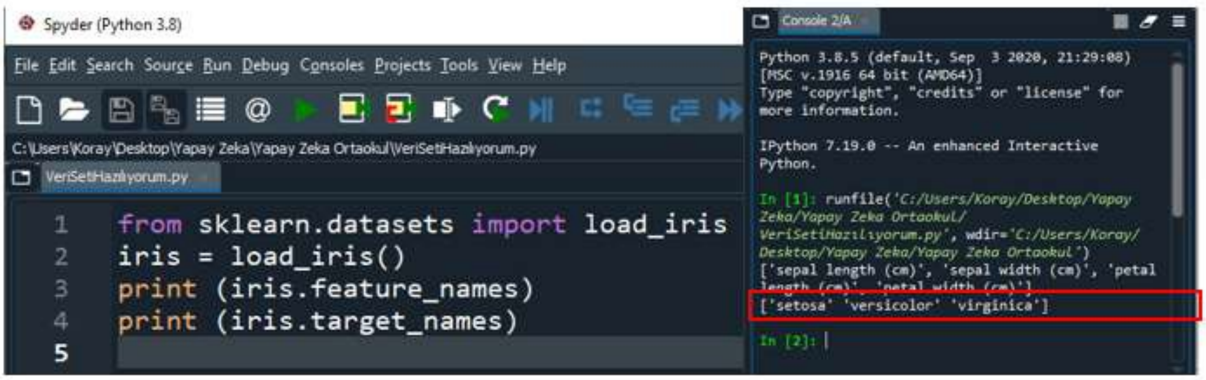
Dikkate alınan veri ön işlemi yapılmış durumdadır. Burada veriler sklearn kütüphanesinden hazır olarak alınmıştır. Birden çok değer alan ve değişim gösteren her şeye "**değişken"** adı verilir. Yapay zekâda sonuca etki eden değişkene **"bağımsız değişken**", başka bir değişkene bağlı olan yani etkilenen değişkene "**bağımlı değişken"** denir. Daha sonra bağımlı/bağımsız hedef değişkenin değeri kategorik sınıftan sayısala çevrilmiştir. Sklearn kütüphanesinden alınan veriyi daha iyi anlayabilmek için verimizdeki bağımsız değişkenlerin (nitelikler) isimleri Şekil 1.16a'da verilen kod satırı kullanılarak elde edilir. Şekil 1.16b'de dört bağımsız değişkenin isim listesi görülmektedir.



(a) (b)

Şekil 1.16. Veri analizi a) bağımsız değişkenleri görüntüleme kodu b) bağımsız değişkenler görüntüsü

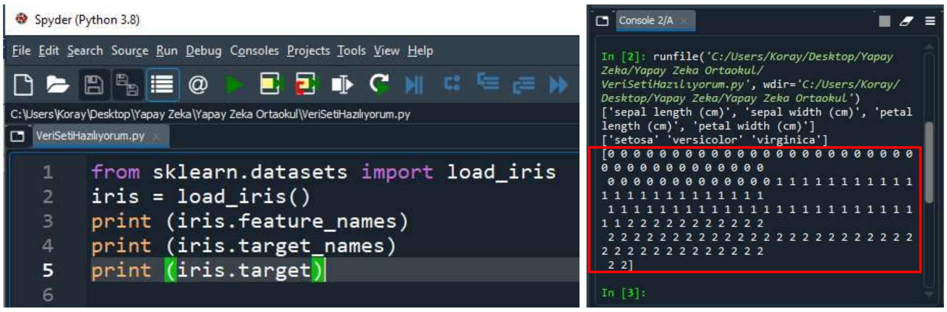
Sklearn kütüphanesinden alınan veriyi daha iyi anlayabilmek için verimizdeki bağımlı değişkenlerin isimleri Şekil 1.17a'da verilen kod satırı kullanılarak elde edilir. Şekil 1.17b'de bağımlı değişkenlerin isimleri listesi görülmektedir.



(a) (b)

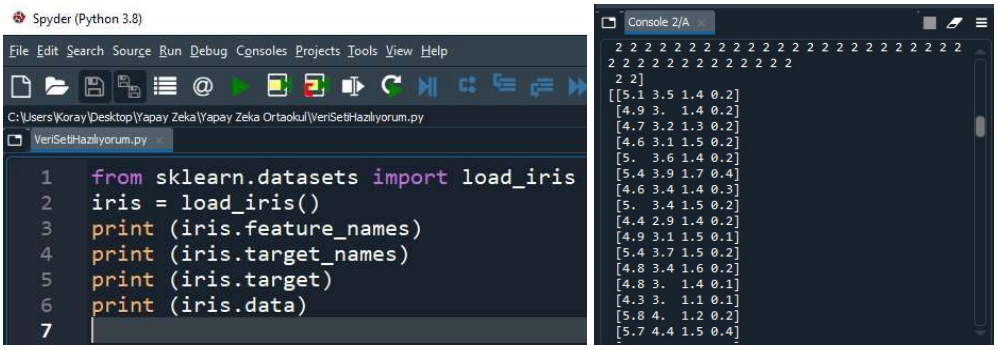
Şekil 1.17. Veri analizi a) bağımlı değişkenleri görüntüleme kodu b) bağımlı değişkenler görüntüsü

Öğrenciler iris çiçek türüne ait üç farklı sınıfı listeler. Şekil 1.18'de bağımlı değişkenlerimiz olan setosa için "0", versicolor için "1" ve virginica için "2" ile kodlanarak sayısallaştırılmıştır. Şekil 1.18a'da iris veri kodu, Şekil 1.19b'de bağımsız değişkenlerin sayısal görüntüsü verilmiştir.



(a) (b)

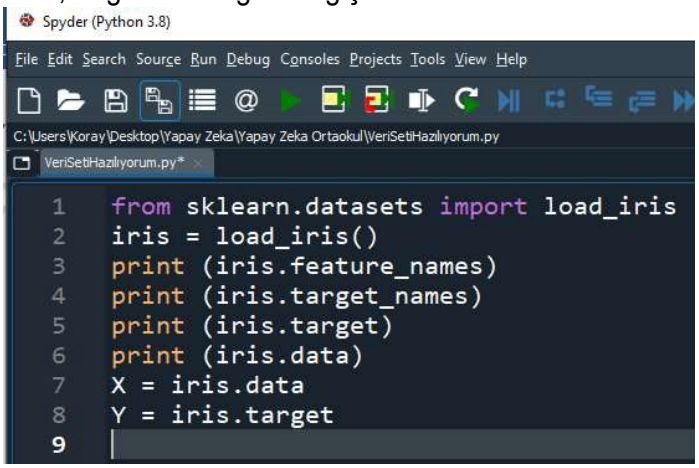
Şekil 1.18. Veri analizi a) bağımlı değişkenleri sayısallaştırma kodu b) görüntüsü



(a) (b)

Şekil 1.19. Veri analizi a) bağımsız değişkenleri sayısallaştırma kodu b) görüntüsü

Veri seti hazırlıyorum son aşamasında öğrenciler bağımlı ve bağımsız değişkenleri oluşturmak için bağımsız nitelikleri X, bağımlı niteliği Y değişkenine atar.



Şekil 1.20. Bağımlı ve bağımsız değişkenleri X ve Y değişkenine atama kod satırı

Dünya'dan Haberler

Birbirine bağlı bir dizi yapay sinir ağı sistemi, ABD'nin Brooklyn şehrinden New Orleans şehrine giderken yolda '1 the Road' adlı bir roman yazdı.

Her şeyi geride bırakıp ülkenin bir ucundan öbür ucuna bir yol gezisine çıkmayı ve dönüşte de o uzun zamandır yazmayı planladığınız romanı yazmış olmayı hayal ettiniz mi? Ettiyseniz de geç kaldınız çünkü artık o işi de robotlar yapıyor.

Birbirine bağlı bir dizi yapay sinir ağı sistemi, ABD'nin Brooklyn şehrinden New Orleans şehrine giderken yolda '*1 the Road*' adlı bir roman yazdı. Yazılımcı Ross Goodwin; GPS koordinatları, Foursquare verileri ve görsel algılama yazılımına bağlı bir kamera ile düz yazı oluşturabilen bir sistem geliştirdi.

Goodwin yapay sinir ağlarını şiir ve edebi eser veri tabanlarıyla eğitti. Bu yüzden '*1 the Road*'un ifadeleri Siri ya da Alexa'ya göre kulağa daha edebi gelebilir. Bunun sebebi yapay zekânın diğerlerine göre daha sanatsal verilerle beslenmiş olması.

Ayrıca 'yapay zekâ roman yazdı' demek biraz abartılı bir ifade olabilir. Roman yazmak için en azından anlamlı cümleler kurabilmek ve birbiriyle ilişkili düşünceleri art arda sıralayabilmek gerek. Yapay zekâ henüz bu kadar gelişmiş değil. Ama sıradan romanlardan sıkıldıysanız ve deneysel bir şeyler arıyorsanız, yapay zekâ size ilginç örnekler sunabilir (Web Kaynağı 1 3)

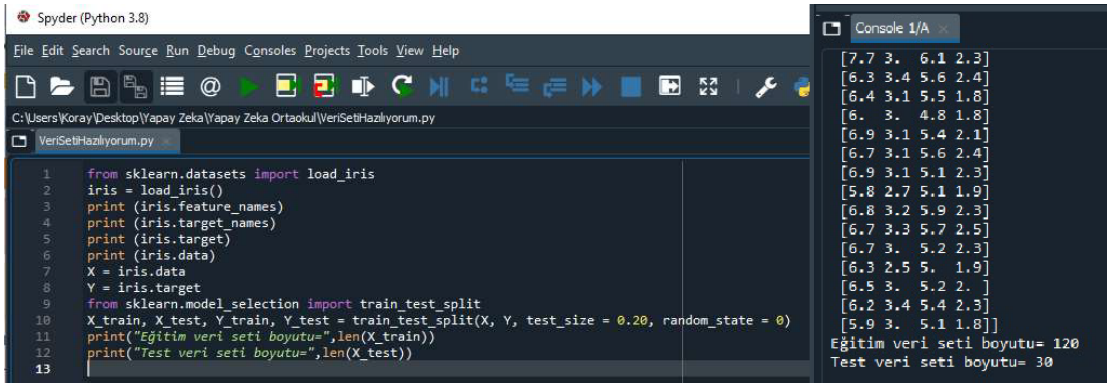
3. HAREKETE GEÇ

3.1. Eğitim ve Test Verilerini Ayırma

Öğrenciler iris çiçeği türüne ait verilerin %80'ini eğitim için, %20'sini ise test için ayırır. Şekil 1.21a'da gösterildiği gibi veri setlerini ayırmak için Python kodlarını yazar. Şekil 1.21b'de ekran görüntüsü verilmiştir.

Eğitmene Not

Eğitmen, öğrencilere veri setini eğitim ve test verileri olarak ayırmayı gösterirken, eğitim veri setinin test veri setinden daha büyük olması gerektiğini belirtir. Eğer test ve eğitim veri seti yüzdesel olarak birbirine yakın olursa modelden elde edilen sonuçların yanlış olabileceğini belirtir. Bu nedenle eğitim veri seti genellikle %75'den başlayarak %80, %85'e kadar ayrılırken test veri seti ise %15 ile %25 arasında değişir. Modelde eğitim ve test verileri yüzdeleri kullanılan veri setine bağlı olarak değişebilmektedir.

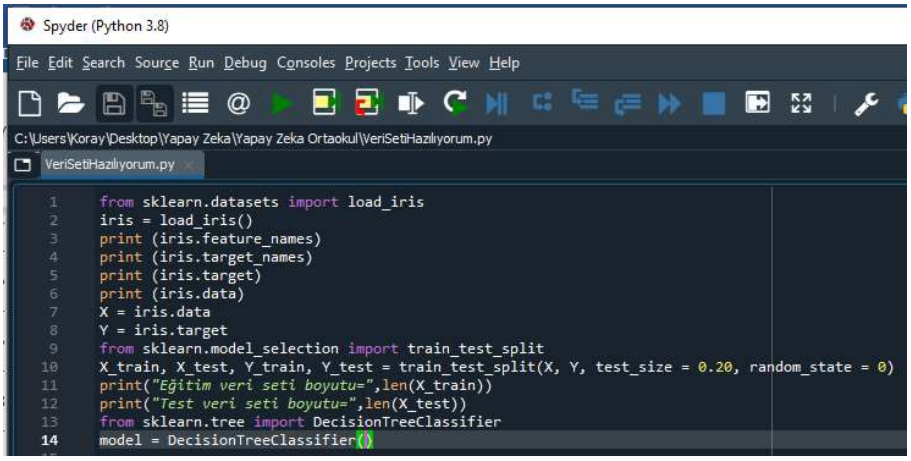


(a) (b)

Şekil 1.21. Eğitim ve test verileri ayırma a) kod b) görüntüsü

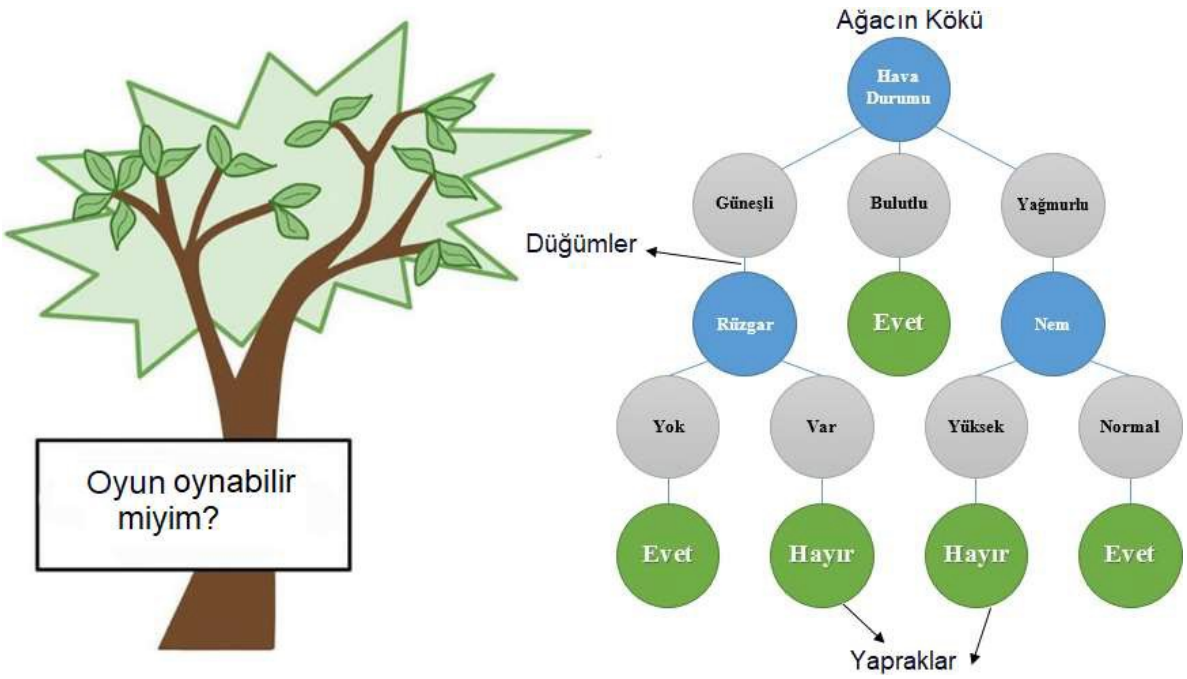
3.2. Model Kurma

Öğrenciler model için scikit-learn kütüphanesinden karar ağaçları sınıflandırıcısını çağırır ve model adında bir değişkene aktarır. Şekil 1.22'de verilen karar ağaçlarına ait algoritmanın Python kodunu editöre yazar.



Şekil 1.22. Karar ağaçları ait algoritmanın kodu

Şekil 1.23'te gösterildiği gibi havanın durumuna göre öğrencilerin okulun bahçesinde oyun oynayıp oynayamayacağını belirlemek için karar ağaçları yapısı gösterilmiştir. Burada karar ağacının algoritmasının kökü, hava durumudur. Karar ağacının düğümleri ise güneş, bulut, yağmur, rüzgâr ve havanın nemi de koşul ifadeleridir. Karar ağacının yaprakları ise okulun bahçesinde oyun oynayıp oynamayacağını belirtir. Ağaç yapısında dallardaki birleşme noktaları (düğümler) bir özelliğe (özniteliğe) tekabül etmekte, her bir dal ise bir kararı işaret etmektedir. Karar ağacında düğümler, sınıflandırılacak gruptaki özellikleri temsil eder ve dallar ise düğümlerin alabileceği değerleri temsil eder.



Şekil 1.23. Karar ağaçlarının yapısı (Web Kaynağı 1.1).

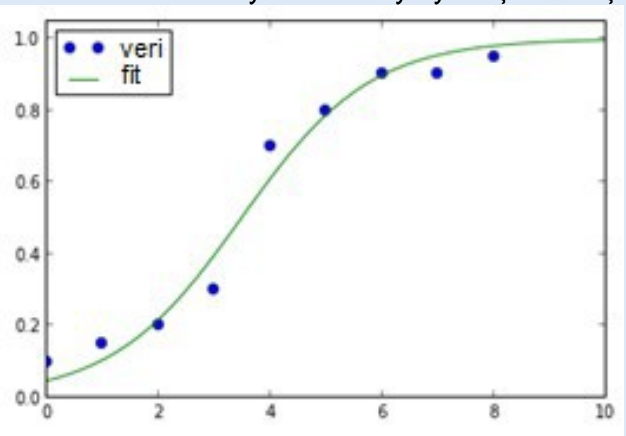
4. YÜRÜT

4.1. Modeli Eğitme

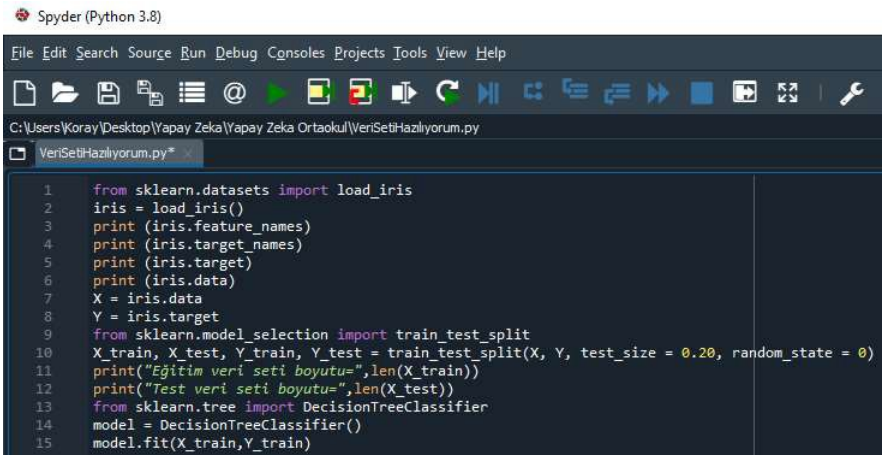
Öğrenci, Şekil 1.24'te gösterildiği gibi Python programlama editöründe iris çiçeği türlerine ait eğitim verilerini **model.fit** komutu ile eğitir.

Eğitmene Not

Eğitmen, öğrencilere Şekil 1.25'te gösterildiği gibi, eğri üzerinden .fit komutunu anlatır. Fit eğrisi üzerinde verileri mümkün olan en yakın noktaya yerleştirmek için model eğitilir.



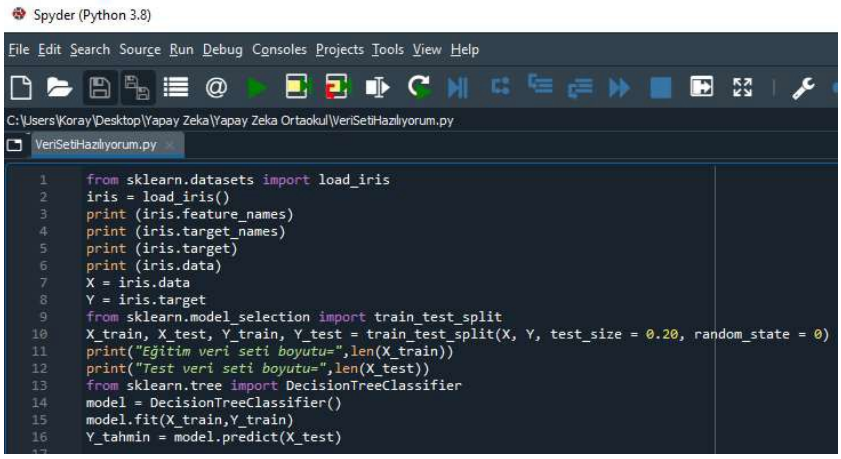
Şekil 1.24. Fit eğrisi



Şekil 1.25. Modeli eğitme

4.2. Model Tahmini

Öğrenci, Şekil 1.26'da gösterildiği gibi Python programlama editöründe iris çiçeği türlerine ait test verilerinin **model.predict** komutu ile giriş verilerine göre çiçeğin hangi tür çiçek sınıfına ait olduğunu tahmin eder.

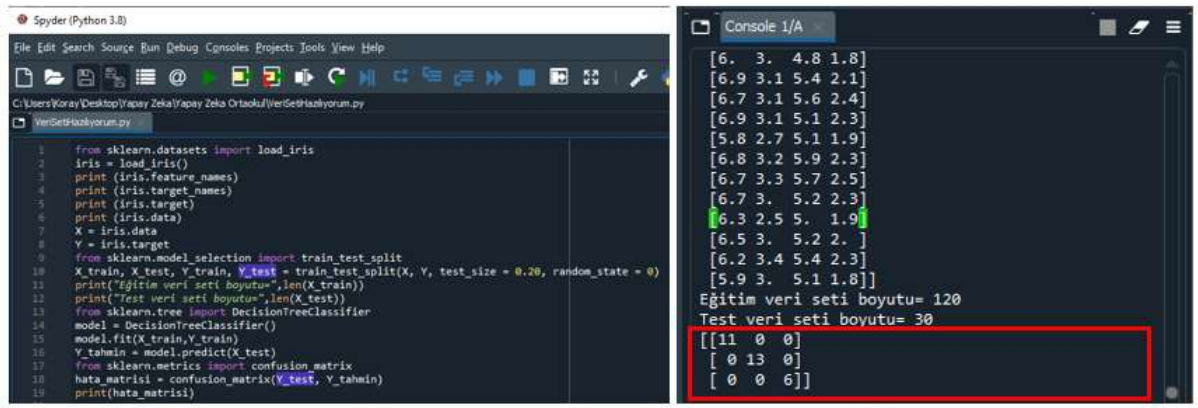


Şekil 1.26. Modeli tahmin etme

5. KARAR VER

5.1. Tahmin-Test Sonuçlarını Karşılaştırma

Öğrenciler, https://github.com/deneyapyz/ortaokul/ adresi altındaki Hafta1 klasöründe yer alan iris.xlsx dosyasını indirip inceler. Burada amaç, karar ağaçları modeli ile eğitilen iris çiçeği sınıfını hata matrisi (confusion matrix) kullanarak modelin başarısını ölçmektir. Hata matrisi yapay zekâ sınıflandırmaları için performans ölçen bir araçtır. Şekil 1.27a'daki Python programlama editörüne hata matrisine ait kodları yazar (ya da Github Hafta1 klasörü altından H1_karar_agaci.py dosyası indirilir). Şekil 1.27b'de görüldüğü gibi hata matrisine ait sonuçlar gözlemlenir.

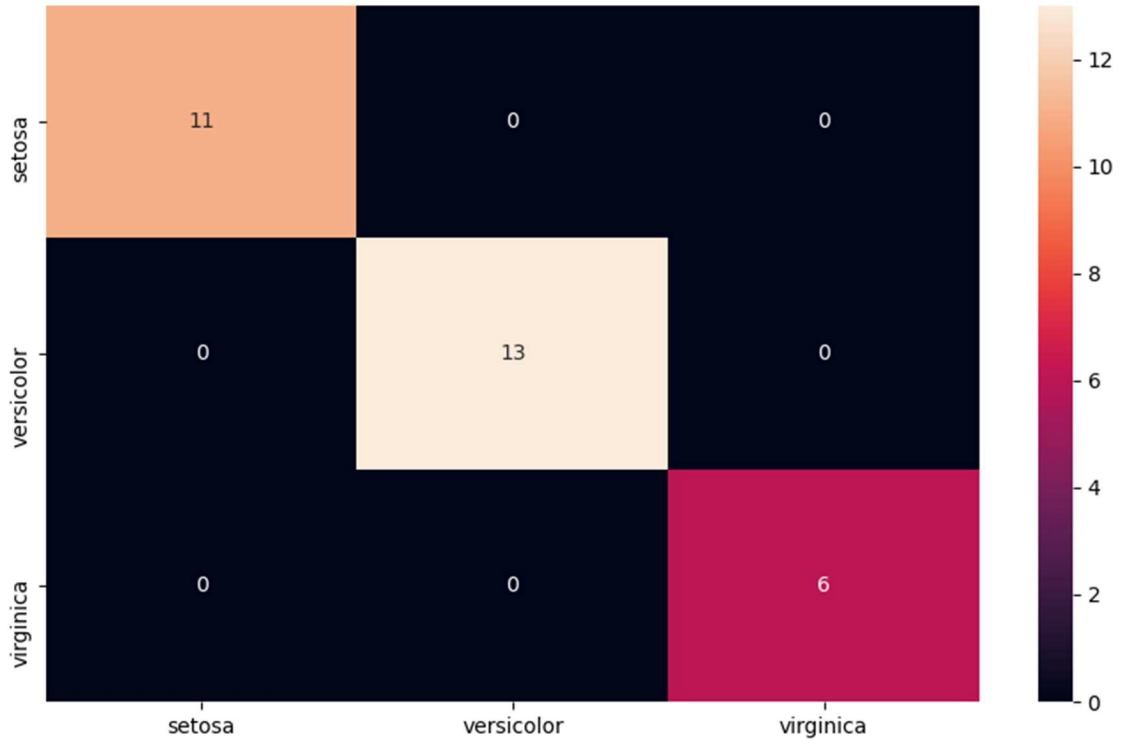


(a) (b)

Şekil 1.27. Hata matrisi a) kod b) sonuç görüntüsü

5.2. Hata Matrisini Değerlendirme

Öğrenciler, iris çiçeğine ait toplam 150 kayıt verinin %80'ini (120) eğitim, %20'i (30) test eğitim seti olarak ayırmıştı. Karar ağaçları modelini 120 kayıt ile eğitmişti. Geriye kalan 30 kayıt (X_test) ise modeli tahmin etmek için kullanıldı (Y_tahmin). Öğrenciler, Şekil 1.28'de gösterildiği gibi 30 test kaydına ait gerçek sonuçlar (Y_test) ile tahmin sonuçlarını karşılaştırdı ve hata matrisi elde etti. Öğrenci matristeki sayıların toplamının test verisi olan 30'a eşit olduğunu gördü.



Şekil 1.28. Hata matrisi

Öğrenciler, hata matrisini incelediğinde 30 adet test kaydında Setosa sınıfına ait 11 tane kayıt var olduğunu ve hepsinin doğru tahmin edildiğini görmüştür. Ayrıca 13 tane versicolor ve 9 tane virginica var olduğunu ve bunların hepsinin de başarıyla doğru tahmin edildiğini görmüştür.

Biliyor musunuz?

İris çiçeği için kullanılan çözüm adımlarını atmosfer olaylarının tahmininde, hastalık teşhisinde veya bir müşterinin satın alabileceği ürünün tahmininde kullanabilirsiniz. Tek yapmanız gereken problemi modelleyip veri setini hazırlamak!

6. UYGULAMANIN PYTHON KODLARI

```
from sklearn.datasets import load_iris
iris = load_iris()
print (iris.feature_names)
print (iris.target_names)
print (iris.target)
```

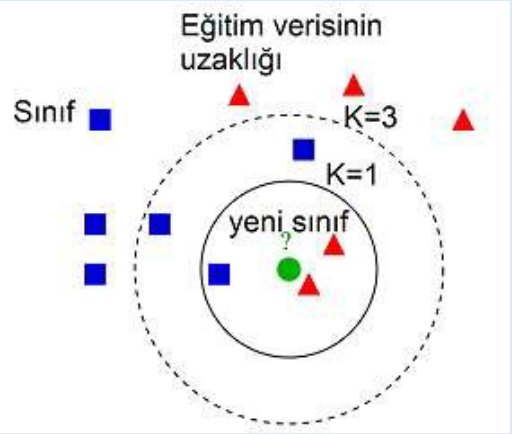
```
print (iris.data)
X = iris.data
Y = iris.target
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.20, random_state = 0)
print("Eğitim veri seti boyutu=",len(X_train))
print("Test veri seti boyutu=",len(X_test))
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
model.fit(X_train,Y_train)
Y_tahmin = model.predict(X_test)
from sklearn.metrics import confusion_matrix
hata_matrisi = confusion_matrix(Y_test, Y_tahmin)
print(hata_matrisi)
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
index = ['setosa','versicolor','virginica']
columns = ['setosa','versicolor','virginica']
hata_goster = pd.DataFrame(hata_matrisi,columns,index)
plt.figure(figsize=(10,6))
sns.heatmap(hata_goster,  annot=True)
```

7. İLAVE ETKİNLİK

Öğrenciler, iris veri çiçeği türlerine ait veri setini kullanarak k-en yakın komşu algoritmasının (k-NN) eğitimini gerçekleştirir. Bu ilave etkinlik ile öğrenciler farklı bir yapay zekâ algoritması kullanarak uygulama gerçekleştirmek için gereken adımları sıralamaları gerektiğini kavrarlar (Bu etkinliğe ait kodlar Github Hafta1 klasörü altında, H1_kNN_algoritmasi.py dosyasında yer almaktadır).

Eğitmene Not

Eğitmen, öğrencilere k-NN algoritmasını şu görsel vasıtasıyla ifade eder:



Şekil 1.29. Fit eğrisi

PYTHON KODLARI

```
from sklearn.datasets import load_iris
iris = load_iris()
print (iris.feature_names)
print (iris.target_names)
print (iris.target)
print (iris.data)
X = iris.data
Y = iris.target
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.20, random_state = 0)
print("Eğitim veri seti boyutu=",len(X_train))
print("Test veri seti boyutu=",len(X_test))
from sklearn.neighbors import KNeighborsClassifier
model =  KNeighborsClassifier ()
model.fit(X_train,Y_train)
Y_tahmin = model.predict(X_test)
from sklearn.metrics import confusion_matrix
hata_matrisi = confusion_matrix(Y_test, Y_tahmin)
```

```
print(hata_matrisi)
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
index = ['setosa','versicolor','virginica']
columns = ['setosa','versicolor','virginica']
hata_goster = pd.DataFrame(hata_matrisi,columns,index)
plt.figure(figsize=(10,6))
sns.heatmap(hata_goster,  annot=True)
```

Düşün, tartış…

Yapay Zekâ ile problem çözmek için gerekenler, modelleme ve veri seti gibi görünüyor… Peki, veri içeriği ve muhtemel siber saldırılar güvenli bir Yapay Zekâ sistemi elde edebilmemiz için önemli midir?

Kaynakça

O'Reilly. (2021). Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems.

Yılmaz, D. Ö. Ü. A., (2021). Yapay Zekâ. Kodlab yayın dağıtım yazılım ltd. Şti.

Web Kaynağı 1.1: https://erdincuzun.com/makine_ogrenmesi/decision-tree-karar-agaci-id3-algoritmasi-classification-siniflama/

Web Kaynağı 1.2: https://www.mshowto.org/veri-on-isleme-nedir.html

Web Kaynağı 1.3:https://www.dunyahalleri.com/yapay-zekâ-gozunden-yol-hikayeleri/

2. Hafta: Yapay Zekâ Matematiği ve Bulanık Mantık

Ön Bilgi:

- Python ile yapay zekâ mantığı ve veri organizasyonu işlemleri gerçekleştirilecektir.
- Python kodlamada karşılaştırma, fonksiyonlar ve kütüphane kullanımı (Bu konuda Python anlatım videoları: 7. Karşılaştırma Op., Mantıksal Operatörler. 8. If Yapısı, 10.-11. Döngüler, 13. Fonksiyonlar, 16. Python Yapay Zekâ kütüphaneleri yardımcı araç olarak kullanılabilecektir).

Haftanın Kazanımları:

- Öğrenciler "yapay zekâ" ekseninde bulanık mantık tabanlı matematiksel işlemleri gerçekleştirir.
- Öğrenciler, yaya geçiti trafik sinyalizasyonuna ait parametrelere göre üyelik fonksiyon sayıları, isimleri, alt ve üst limitleri belirler.
- Öğrenciler Python programlama dilinde scikit-fuzzy kütüphanesindeki bulanık mantık modellemesi için gerekli olan "**antecedent","arrange","consequent", "ControlSystem", "ControlSystemSimulation" ve "compute"** komutlarını uygulayarak öğrenir.
- Öğrenciler, yaya geçidi trafik sinyalizasyonunda yaya sayısı değerini bulanık mantık modeli ile ("az", "orta" ve "yüksek" şeklinde) organize eder ve ilgili kurallar yardımıyla desteklenen sistemi Python programlama dilinde oluşturur.
- Öğrenciler yaya geçidi trafik sinyalizasyonu örneğinde öğrendiği bulanık mantık komutlarını kullanarak "otomatik fren sistemi" örneğini uygulayarak sentezler.
- Öğrenciler Python ile bulanık mantık modelleme ve çözüm üretmeyi uygular.

Haftanın Amacı:

Bu haftanın amacı, "yapay zekâ matematiği" ve "bulanık mantık" kavramlarının tüm öğrenciler tarafından anlaşılmasını sağlamaktır. Haftanın bir diğer amacı, bulanık mantık kullanılarak farklı örnekler ile öğrencilerin etkinliklere yönelik ilgilerini artırmaktır. Ayrıca, öğrencilere Python ile bulanık mantık çözüm üretme yetenekleri kazandırılacaktır.

Kullanılacak Malzemeler:

Bilgisayar, kâğıt, kalem, Python kod editörü, örnekler için çıktı görselleri

Haftanın İşlenişi:

Algıla: Yapay zekâ matematiği ve bulanık mantık sistemi kavramlarını öğrenir.

Tasarla: Yapay zekâda veri organize edebilir.

Harekete Geç: Python ile eğitim ve test verilerini ayırma, model kurma gibi süreçleri gerçekleştirebilir.

Yürüt: Eğitmenin, öğrencilere Python ile model tasarlama işlemlerini gerçekleştirebilmeleri için uygulamalarına birebir ve etkili rehberlik eder.

Karar Ver: Haftanın içeriğinin Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşimli bir biçimde kullanılır.

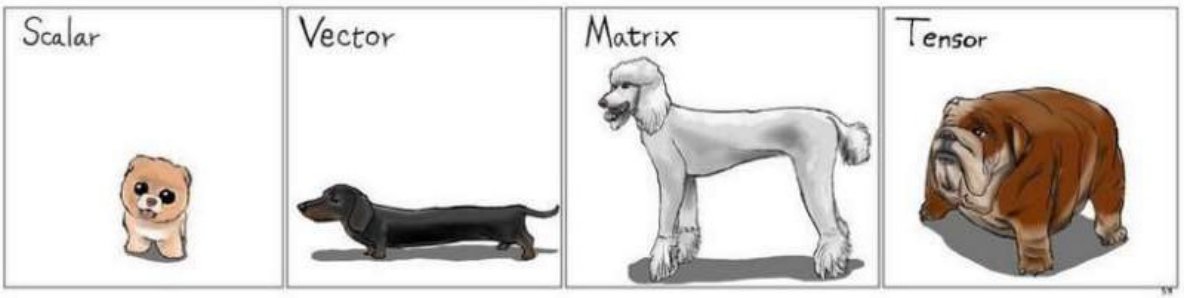
1. ALGILA

1.1. Mantık

Matematiksel becerileri kazanabilmek için matematiksel düşünme tarzını geliştirerek uygulamaya koymak mantığın temelini oluşturur (Korkmaz, 2019). Mantık, insan zihni gibi doğruyu ve yanlışı ayırt etmek için kullanılan bir düşünme disiplinidir. Mantık kavramı doğru (1) veya yanlıştan (0) oluşan bir yargının sonucudur. Örneğin; "İstanbul, Türkiye'nin doğusundadır." cümlesi bir önermedir ve bu önerme yanlıştır. "23 Nisan, Ulusal Egemenlik ve Çocuk Bayramı'dır." önermesi ise doğru bir önermedir. Mantık kavramı insan beyninin problem çözme teknikleri ile oldukça benzerdir. İnsan beyni ilk olarak problemin kaynağı olan faktörleri tespit ettikten sonra, problemin çözümü için çözüm aşamalarını mantıksal olarak gerçekleştirir.

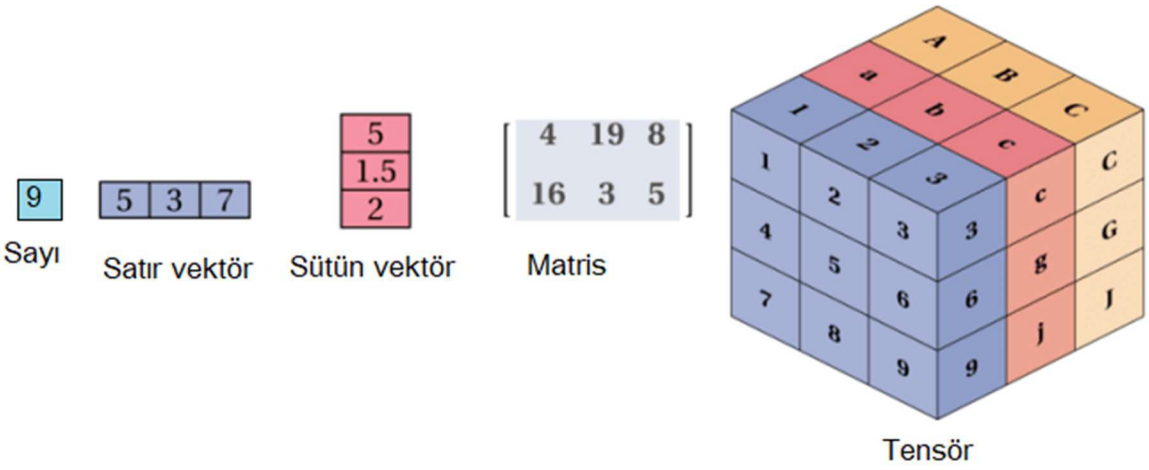
1.2. Python ile Yapay Zekâ Matematiği

Öğrenciler ilk haftada öğrendikleri Python ile numpy, pandas ve scipy yapay zekâ kütüphaneleri ile basit bir şekilde matematiksel işlemleri gerçekleştirir. Python ile yapay zekâ matematiği sayılar, matrisler, vektörler, tensörler gibi kavramları kapsar (Şekil 2.1).



Şekil 2.1. Skalar, vektör, matris ve tensör görselleri (Web Kaynağı 2.1)

İnsanın yaşını tanımlarken kullanılan sayılar ifadesi boyutsuz olarak kullanılır. İnsanın yaşı yerine fotoğrafı ise 2 boyutlu sayılardan oluşan matris ile ifade edilir (Şekil 2.2). Matrislerin satır ve/veya sütunlarını oluşturan her bir boyutuna vektör denir. Şekil 2.2'de gösterildiği gibi tensör kavramı ise, insanın damarındaki kanın akışı ile, damarın hacimsel ve yüzeysel alan değişimi ifade edilirken kullanılır. Örneğin zekâ küpü tensöre bir örnek olarak verilebilir.



Şekil 2.2. Yapay zekâ matematiğindeki veri türleri

1.3. Bulanık Mantık Tekniği

Bulanık mantık insan düşüncesini fonksiyonlara dönüştüren hesaplamalı matematiksel işlemlerdir. Bulanık mantık kavramı klasik yanlış (0) ve doğru (1) mantığı arasında derecelendirme yaparak kümelendirme işlemini gerçekleştirir. Problemin çözümü esnasında bazı önermelerin doğru ya da yanlış olarak değerlendirilmesi mümkün olmayabilir. Bu durumun sebebi, önermelerin doğru bir şekilde ölçülememesinden kaynaklanmaktadır. İnsan beyninde gerçek durumlar daha çok ara değerler ile temsil edilebileceği için klasik (keskin) mantığa karşı **"bulanık mantık"** kavramı öne sürülmektedir. Öğrenciler bulanık mantık ile modelleme işleminde, giriş parametreleri ile çıkış parametrelerini ilişkilendirerek küme oluştur ve bulanık mantık kurallarını tanımlar. Örneğin bir kişinin yaşını tahmin etmeye çalışalım. Kişinin yaşını görsel olarak anlamak kesin bir sonuç değildir. Kişinin kesin yaşını belirlemek için ya kimliğine bakılır ya da tıbbi testler uygulanır. Bu sonuçlara bakarak, kişinin yaşı hakkında net bir bilgi elde edilir. Örneğin "Bu kişi 18 yaşından büyüktür." gibi bir önermeye yüzde yüz yanlış ya da yüzde yüz doğru gibi bir yorum getirmek mümkündür. Ancak önerme "Bu kişi gençtir." şeklinde ortaya atılırsa bulanık mantık kavramı ortaya çıkacaktır. Şekil 2.3'te bulanık mantık işlem sırası gösterilmiştir. Yapay zekâda bulanık mantık bileşenleri şöyledir:

Giriş: Yapay zekâ bulanık mantık sistemlerinde ilk olarak veri girişi yapılır. Böylece bulanık mantık sistemi çalışmaya başlar.

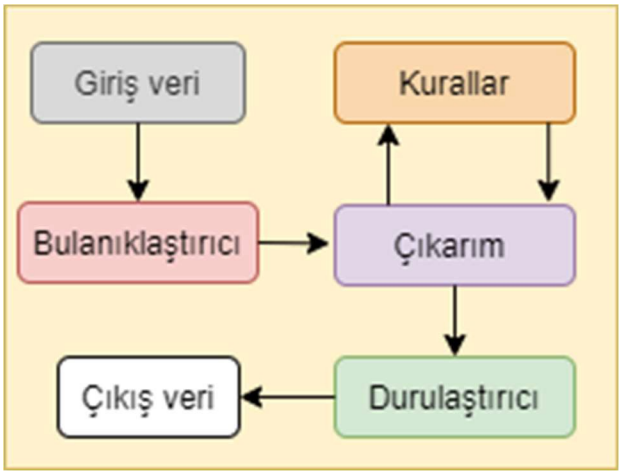
Bulanıklaştırıcı: Bulanık mantık sistemine girilen verileri 0 ile 1 arasındaki bulanık değerlere dönüştürür.

Kurallar: Bulanıklaştırma işlemi sırasındaki çıkarım kuralları belirlenir.

Çıkarım: Bulanık kuralları kullanarak bulanık giriş değerleri için bulanık sonuçlar çıkarır.

Durulaştırıcı: Çıkarımdan elde edilen sonuçları istenilen gerçek değerlere dönüştürür.

Çıkış: Bulanık mantık sisteminden elde edilen çıkış verisidir.



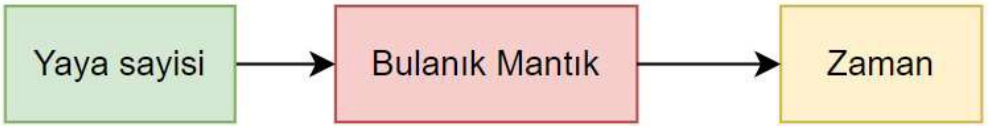
Şekil 2.3. Bulanık mantık sistemin genel görüntüsü

2. TASARLA

Öğrenciler, ilk olarak bulanık mantık ile trafik sinyalizasyon problemini anlar. Şekil 2.4'te gösterilen kavşaktaki yaya yoğunluğuna bağlı olarak trafik lambaların sürelerini bulanık mantık ile modeller.

Şekil 2.4. Yaya geçiti trafik sinyalizasyonu

Şekil 2.5'te gösterildiği gibi öğrenciler ilk olarak bulanık mantık giriş çıkış parametresini belirler.



Şekil 2.5. Sistemin giriş ve çıkış parametrelerin belirlenmesi

Yaya trafik sinyalizasyonuna ait bulanık model ile yaya sayısına göre en uygun yeşil ışık yanma süresi belirlenmesi amaçlanmıştır.

Düşün, tartış…

Günlük hayatta kullandığımız makinelere Bulanık Mantık eklemek suretiyle onları daha verimli hale getirebilir miyiz? Örneklerle tartışalım!

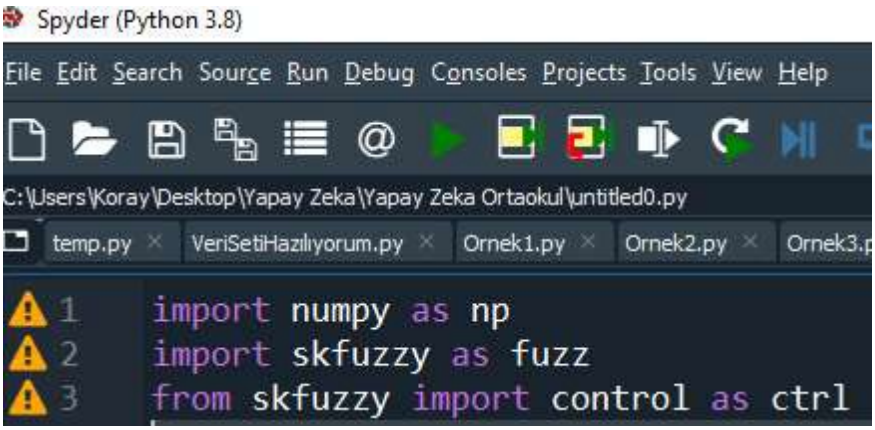
3. HAREKETE GEÇ

Öğrenciler yaya trafik sinyalizasyonuna ait giriş-çıkış parametrelerine göre tüm parametrelerin üyelik fonksiyon sayılarını, isimlerini, alt ve üst limitlerini belirleyerek Python kodlarını hazırlar. Öğrenciler, ilk olarak, Şekil 2.6'da gösterilen Anaconda Prompt (Anaconda 3) uygulama ekranından scikit-fuzzy kütüphanesini yükler.

```
C:\Users\Koray>pip install scikit-fuzzy
```

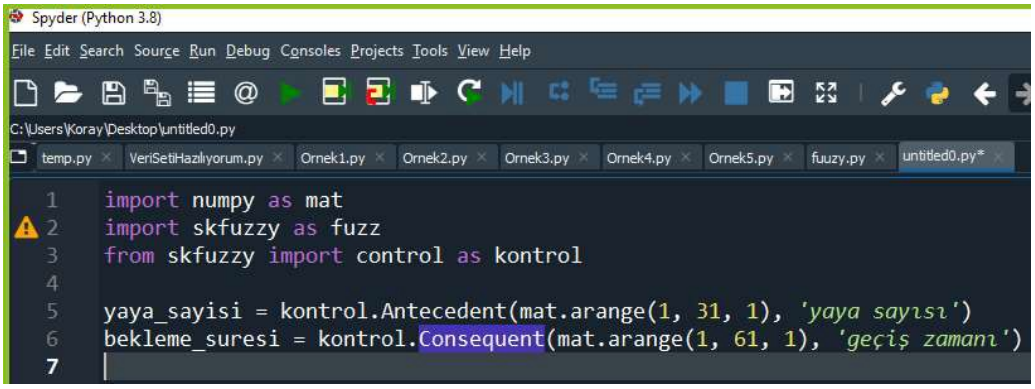
Şekil 2.6. Scikit-fuzzy kütüphanesi

Şekil 2.7'deki bulanık mantık (fuzzy) kütüphaneleri kodlarını yazar. 2. ve 3. satırda yer alan bulanık mantık kütüphanesini editöre yükler.



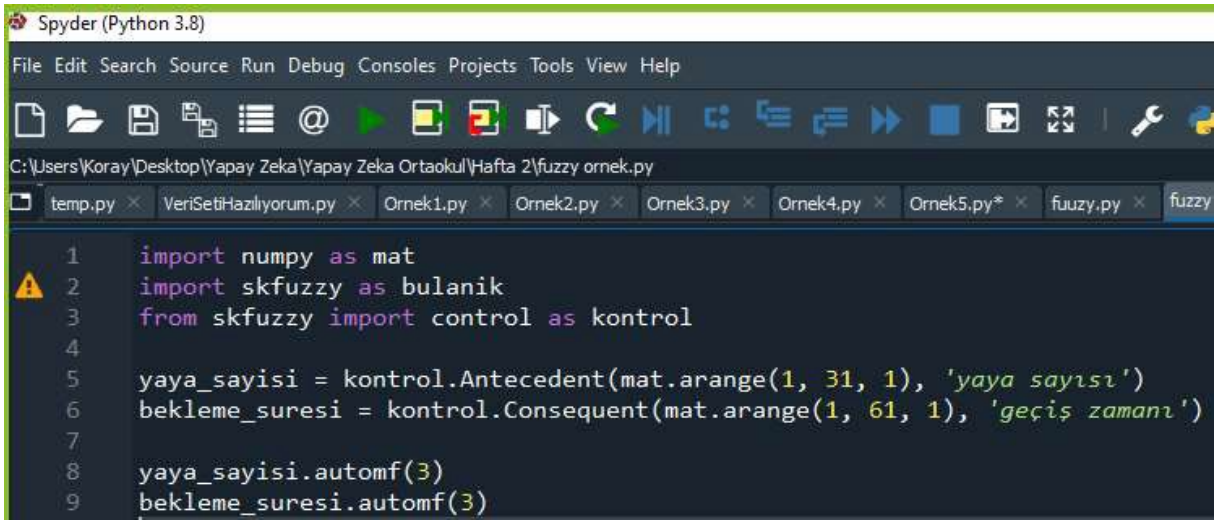
Şekil 2.7. Yaya trafik sinyalizasyonu için Python kütüphanelerin kod satırı

Öğrenciler, kütüphane kodlarını yazdıktan sonra giriş parametresi olan yaya sayısını 0 ile 30 aldığında çıkış parametresinin geçiş zamanını 0 ile 60 saniye arasında belirler. Bunun için giriş parametrelerini belirlemede bulanık mantık kontrol sistemlerinde giriş değişkeni olan "**kontrol.antecedent**" komutunu kullanarak "**mat.arrange"** ile giriş aralığını belirler. Çıkış parametresini belirlerken bulanık mantık kontrol sistemlerinde çıkış değişkeni olan **"kontrol.Consequent"** ile yine **"mat.arrange"** komutunu kullanarak çıkış aralığını belirler.



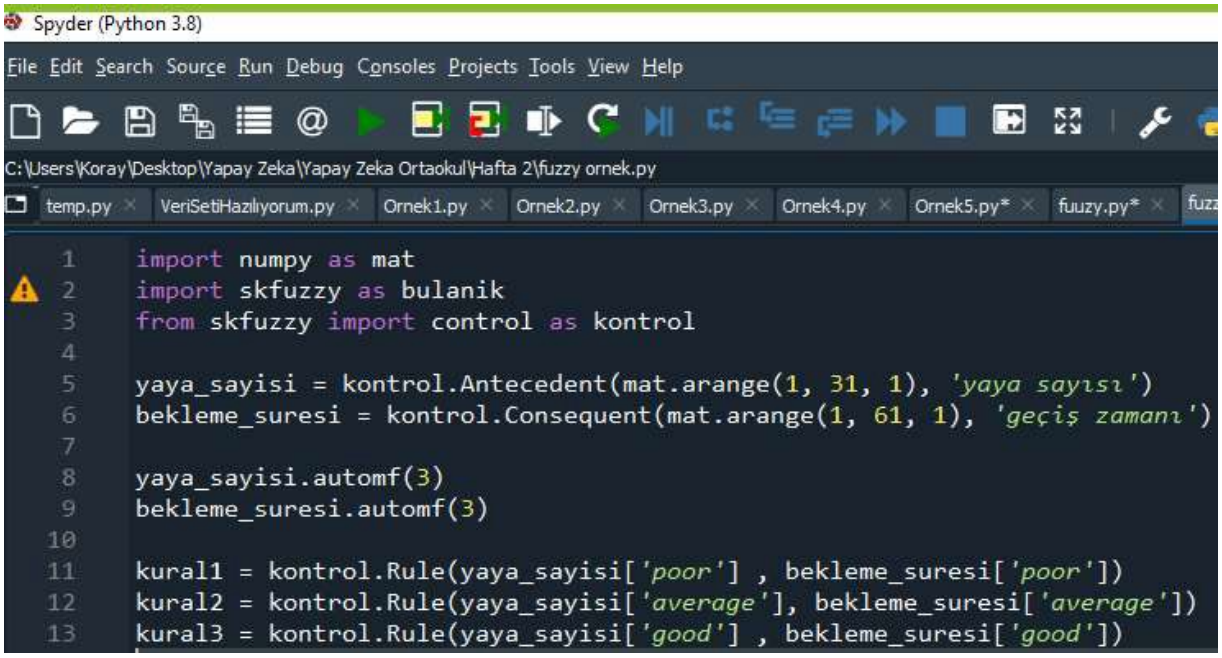
Şekil 2.8. Bulanık mantık kontrol sistemlerinde giriş ve çıkış parametrelerin belirlenmesi

Öğrenciler, üyelik fonksiyon kümelerini oluştururken, giriş-çıkış değerlerinden kaçının "sözel giriş/çıkış değeri" ile isimlendirileceğine karar verir. Öğrenciler, Şekil 2.9'da gösterildiği gibi "automf(3)" komutunu kullanarak giriş parametresi için yaya sayısını "poor(az)", "average(orta)", "good(yüksek)" şeklinde adlandırılarak sisteme tanıtır.



Şekil 2.9. Dilsel değişken isimlendirilmesi

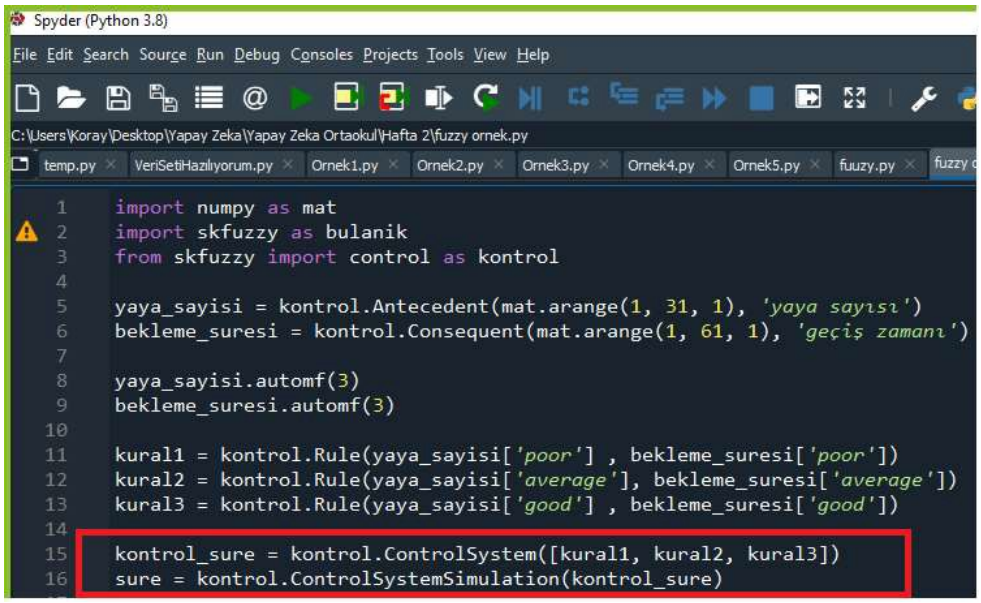
Öğrenciler bulanık kontrol sisteminin kurallarını Şekil 2.10'da gösterildiği gibi oluşturur. Kural-1'de yaya sayısı poor (az) olduğunda bekleme süresi poor (az), Kural-2'de yaya sayısı average (orta) olduğunda bekleme süresi average (orta), Kural-3'te yaya sayısı good (yüksek) olduğunda bekleme süresi good (yüksek) olmak üzere üç adet kural belirlenmiştir.



Şekil 2.10. Kuralların kod satırı

4. YÜRÜT

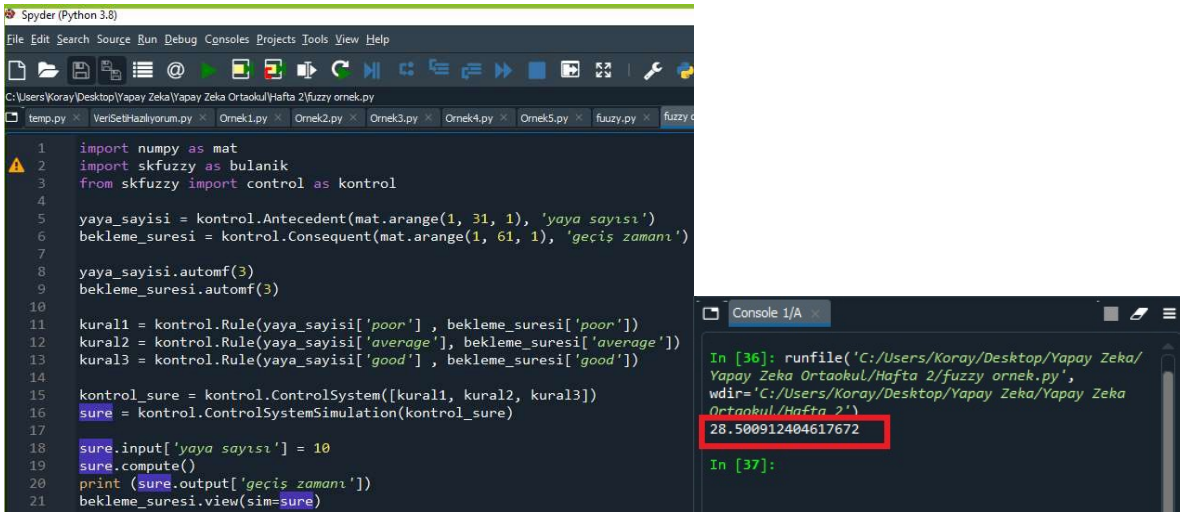
Öğrenciler trafik sinyalizasyonu için oluşturulan “az”, “orta” ve “yüksek” yaya sayıları için bekleme sürelerini yine “az”, “orta” ve “yüksek” olmak üzere gerçekleştirilen üyelik fonksiyonlarını bulanık mantık ile modellemek için Python kodlarını yazar. Şekil 2.11’de ekran görüntüsü verilmiştir. Öğrenciler, ilgili üç kurala göre “ControlSystem” komutu kullanarak basit bir bulanık mantık kontrol sistemi oluşturur. Oluşturulan bulanık kontrol sisteminin simülasyonu için “ControlSystemSimulation” komutunu kullanır.



Şekil 2.11. Bulanık kontrol sisteminin oluşturulması ve simülasyon kod satırı

5. KARAR VER

Öğrenciler, üç kurala göre oluşturmuş oldukları bulanık mantık modelinin sonuçlarına Şekil 2.12'de verilen kod satırlarını kullanarak karar verir (İlgili koda Github platformu: https://github.com/deneyapyz/ortaokul/ kapsamında yer alan, Hafta2 klasörü altındaki H2_bm_trafik-sinyalizasyon.py adlı dosyadan da erişilebilir). Şekil 2.12a'da "sure.input" komutu ile öğrenci bulanık mantık sistemine göndermek istediği yaya sayısını girer. Daha sonra "sure.compute()" komutu kullanılarak oluşturulan bulanık mantık sisteminde girilen yaya sayısına göre bekleme süresini bulanık mantık modelinde hesaplatır. Şekil 2.12b'de 10 yaya için bulanık mantık modeli geçiş süresini 28,5 saniye olarak hesaplar.



Şekil 2.12. Trafik sinyalizasyon için bulanık mantık modelinin değerlendirilmesi

Eğitmene Not

Eğitmen, verilen yaya trafik sinyalizasyon bulanık mantık model Python kodlarında
sure.input['*yaya sayısı*']= 5
sure.input['*yaya sayısı*']= 15
sure.input['*yaya sayısı*']= 37
gibi farklı örnekler üzerinde bulanık mantık modelinden elde edilen geçiş zamanı sürelerini öğrenciye uygulamalı olarak gösterir.

6. UYGULAMANIN PYTHON KODLARI

```
import numpy as mat
import skfuzzy as bulanik
from skfuzzy import control as kontrol

yaya_sayisi = kontrol.Antecedent(mat.arange(1, 31, 1), 'yaya sayısı')
bekleme_suresi = kontrol.Consequent(mat.arange(1, 61, 1), 'geçiş zamanı')

yaya_sayisi.automf(3)
bekleme_suresi.automf(3)

kural1 = kontrol.Rule(yaya_sayisi['poor'] , bekleme_suresi['poor'])
kural2 = kontrol.Rule(yaya_sayisi['average'], bekleme_suresi['average'])
kural3 = kontrol.Rule(yaya_sayisi['good'] , bekleme_suresi['good'])

kontrol_sure = kontrol.ControlSystem([kural1, kural2, kural3])
sure = kontrol.ControlSystemSimulation(kontrol_sure)

sure.input['yaya sayısı'] = 10
sure.compute()
print (sure.output['geçiş zamanı'])
bekleme_suresi.view(sim=sure)
```

7. İLAVE ETKİNLİK

Öğrenciler, ilk olarak bulanık mantık ile otomatik fren sistemi problemini anlar. Şekil 2.13'te gösterildiği gibi otomatik fren sistemi için bulanık mantık kontrol sistemi giriş parametreleri mesafe ve hız, çıkış parametresi ise uygulanması gereken fren basınç değerlerini bulanık mantık modeli ile belirler. Bu etkinlik kapsamında problem çözümüne ilişkin bulanık mantık sistemi kodlanır (İlgili koda Github platformu Hafta2 klasörü altındaki H2_bm_fren-sistemi.py adlı dosyadan erişilebilir.).

Şekil 2.13. Bulanık mantık ile otomatik fren sistemi

PYTHON KODLARI:

```
import numpy as mat
import skfuzzy as mantik
from skfuzzy import control as kontrol

mesafe = kontrol.Antecedent(mat.arange(0, 50, 1), 'mesafe')
hiz = kontrol.Antecedent( mat.arange(0, 100, 1), 'hiz')
fren_basinci = kontrol.Consequent(mat.arange(0, 100, 1),'fren_basinci')

mesafe['çok yakın'] = mantik.trimf(mesafe.universe, [0, 0, 10])
mesafe['yakın'] = mantik.trimf(mesafe.universe, [5, 15, 25])
mesafe['uzak'] = mantik.trimf(mesafe.universe, [20, 30, 40])
mesafe['çok uzak'] = mantik.trimf(mesafe.universe, [35, 50, 50])

hiz['çok yavaş'] = mantik.trapmf(hiz.universe, [0, 0, 20, 30])
hiz['yavaş'] = mantik.trapmf(hiz.universe, [20, 30, 45, 55])
hiz['hızlı'] = mantik.trapmf(hiz.universe, [45, 55, 70, 80])
hiz['çok hızlı'] = mantik.trapmf(hiz.universe, [70, 80, 100,100])

fren_basinci['çok düşük'] = mantik.trimf(fren_basinci.universe, [0, 20, 40])
fren_basinci['düşük'] = mantik.trimf(fren_basinci.universe, [20, 40, 60])
fren_basinci['yüksek'] = mantik.trimf(fren_basinci.universe, [40, 60, 80])
fren_basinci['çok yüksek'] = mantik.trimf(fren_basinci.universe, [60, 100, 100])
```

```
kural1 = kontrol.Rule(mesafe['çok yakın'] & hiz['çok yavaş'] , fren_basinci['çok yüksek'])
kural2 = kontrol.Rule(mesafe['yakın'] & hiz['çok yavaş'] , fren_basinci['çok düşük'])
kural3 = kontrol.Rule(mesafe['çok yakın'] & hiz['yavaş'] , fren_basinci['çok yüksek'])
kural4 = kontrol.Rule(mesafe['yakın'] & hiz['yavaş'] , fren_basinci['düşük'])
kural5 = kontrol.Rule(mesafe['uzak'] & hiz['yavaş'] , fren_basinci['çok düşük'])
kural6 = kontrol.Rule(mesafe['çok yakın'] & hiz['hızlı'] , fren_basinci['çok yüksek'])
kural7 = kontrol.Rule(mesafe['yakın'] & hiz['hızlı'] , fren_basinci['düşük'])
kural8 = kontrol.Rule(mesafe['uzak'] & hiz['hızlı'] , fren_basinci['çok düşük'])
kural9 = kontrol.Rule(mesafe['çok yakın'] & hiz['çok hızlı'] , fren_basinci['çok yüksek'])
kural10 = kontrol.Rule(mesafe['yakın'] & hiz['çok hızlı'] , fren_basinci['yüksek'])
kural11 = kontrol.Rule(mesafe['uzak'] & hiz['çok hızlı'] , fren_basinci['düşük'])
kural12 = kontrol.Rule(mesafe['çok uzak'] & hiz['çok hızlı'] , fren_basinci['çok düşük'])

fren_kontrol = kontrol.ControlSystem([kural1, kural2, kural3, kural4, kural5,kural6,
                                      kural7, kural8, kural9, kural10, kural11, kural12])
frenleme = kontrol.ControlSystemSimulation(fren_kontrol)

v = int(input("Hızı gir (0-100 km/h) : "))
s = int(input("mesafeyi gir (m) : "))

if (v/2 <= s):
    print ("fren basılması gerek yoktur")
else:
  frenleme.input['hiz'] = v
  frenleme.input['mesafe'] = s

  frenleme.compute()
  print ("fren basıncı (%): ", frenleme.output['fren_basinci'])
  print ("yeni hiz değeri: ", v-(v*frenleme.output['fren_basinci']/100))
```

Biliyor musunuz?

Bulanık Mantık özellikle ev aletlerinde oldukça sık kullanılmıştır. Belki de farkında olmadan bu tür aletleri (örneğin bulaşık makinesi, çamaşır makinesi) evinizde kullanmış olabilirsiniz! Bu özellikteki ev aletlerini bulmak için Web'i tarayın!

Eğitmene Not

Eğitmen, öğrencilerin 1. ve 2. hafta içerisinde elde ettikleri bilgi-becerileri değerlendirmek, aktif katılımlarını sağlamak ve onları gelecek haftalara motive etmek için 'Kahoot' uygulamasını ya da benzeri bir Web 2.0 uygulamasını kullanarak 'bilgi yarışması' düzenleyebilir.

1. ve 2. hafta bağlamında sorulabilecek sorular; Yapay Zekâ kavramının temelleri, veri analiz süreçleri, Makine Öğrenmesi temelleri, değinilen önemli Python kodları, Bulanık Mantık temelleri ve uygulamaları yönünde olabilir.

Kaynakça

Korkmaz, S. (2019). 9. Sınıf Öğrencilerinin Mantık Konusundaki Kavram Yanılgıları (Yüksek Lisans Tezi, Necmettin Erbakan Üniversitesi Eğitim Bilimleri Enstitüsü).

Web Kaynağı 2.1: https://medium.com/zaferdemirkol/herkes-i%C3%A7in-yapay-zekâ-matemati%C4%9Fi-2-vekt%C3%B6r-matris-i%CC%87%C5%9Flemleri-d7c63ad53f9

3. Hafta: Makine Öğrenmesi Kavramı ve Olasılıklı Çözümler İçin Bayes Öğrenmesi

Ön Bilgi:

- Python ile yapay zekâ mantığı, veri organizasyonu, yapay zekâ matematiği öğrenecektir.
- Python kodlamada döngüler, fonksiyonlar ve kütüphane yapılarının kullanımı (Bu konuda Python anlatım videoları: 10-11. Döngüler, 13. Fonksiyonlar, 14-16. Python kütüphaneleri serisi yardımcı araç olarak kullanılabilecektir).

Haftanın Kazanımları:

- Öğrenciler, makine öğrenmesi kavramını ve Bayes öğrenmesi algoritmasını kullanır.
- Öğrenciler, verileri makine öğrenme algoritmaları ile gerçek hayat problemlerine uygular.
- Öğrenciler makine öğrenmesindeki temel aşamaları bilir ve uygulamalarda kullanır.
- Öğrenciler regresyon, sınıflandırma ve kümeleme yöntemlerini uygular.
- Öğrenciler, sonucun sebebini bulurken sonucun hangi olasılıkla hangi sebepten kaynaklandığını veren Bayes Teoremini örneklerle kavrar.
- Öğrenciler Python programlama dilini kullanarak aracın özelliklerine göre satış durumunu tahminleme etkinliği için program kodunu yazabilir.

Haftanın Amacı:

Bu haftanın amacı, "yapay zekâ altdalı olan makine öğrenmesi" ve "Bayes öğrenme" kavramlarının tüm öğrenciler tarafından doğru bir şekilde anlaşılmasını sağlamaktır. Haftanın bir diğer amacı, makine ve Bayes öğrenmeleri kullanılarak, farklı örnekler ile öğrencilerin ilgisini çekerek etkinliklere yönelik ilgilerini artırmaktır. Ayrıca, öğrencilere Python ile makine öğrenme algoritmaları ile örnek modelleme ve çözüm üretme yeteneği kazandırılacaktır.

Kullanılacak Malzemeler:

Bilgisayar, kâğıt, kalem, Python kod editörü, örnekler için çıktı görselleri

Haftanın İşlenişi:

Algıla: Makine öğrenmesi temelleri ve Bayes öğrenme tekniği kavramlarını öğrenir.

Tasarla: Bayes öğrenme ile araç satış probleminin giriş ve çıkış parametrelerini tasarlar.

Harekete Geç: Python ile veri setindeki metinsel değerleri sayısallaştırıp eğitim ve test verileri ile model kurar. Bu hafta için Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşim Harekete Geç aşaması ile başlatılabilmektedir.

Yürüt: Eğitmenler öğrenciler ile etkileşimli bir biçimde Bayes öğrenmesi kullanarak modeli eğitir ve aracın satış tahmin durumunu belirler.

Karar Ver: Haftanın içeriğinin Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşimli kullanımı

1. ALGILA

1.1. Makine Öğrenmesi Temelleri

Günümüzde teknolojinin hızla gelişmesiyle bilgisayar, cep telefonu, tablet gibi akıllı cihazların ve internet teknolojisinin hızla yayılması ile birlikte üretilen veriler artmış ve "büyük veri" kavramı ortaya çıkmıştır. Bunun en önemli nedeni ise hayatımızın her alanında Instagram, Twitter ve Youtube paylaşımları, nesnelerin interneti (IoT) gibi dijital bir hareket olmasıdır. Büyük veri; veri hacmi, veri hızı, veri çeşitliliği, verinin düzensiz olması ve verinin değerli olması olmak üzere beş bileşenden oluşur. Veri hacmi, verinin yüksek hacimli olduğunu belirtir. Örneğin bir uçağın motorunda ve diğer bileşenlerinde milyonlarca sensör mevcuttur. Bu sensörler uçakların yaptığı her bir hareketi anlık olarak toplayarak terabaytlar seviyesinde veri üretmektedir (Şekil 3.1).

Şekil 3.1. Uçak sensörlerinden bir yıl boyunca elde edilen veri boyutu (Web Kaynağı 3.1)

Büyük verinin ikinci bileşeni ise verinin hızıdır. Örneğin Şekil 3.2'de gösterildiği gibi bir dakika içerisinde 200+ milyon e-mail, 4 milyon Facebook beğenmesi, 1+ milyon Instagram beğenmesi, milyonlarca atılan tweetler gibi veriler oldukça hızlı aktarılmaktadır. (Verinin kaynağı ve hangi yıla ait olduğu yazılırsa daha iyi olur.)

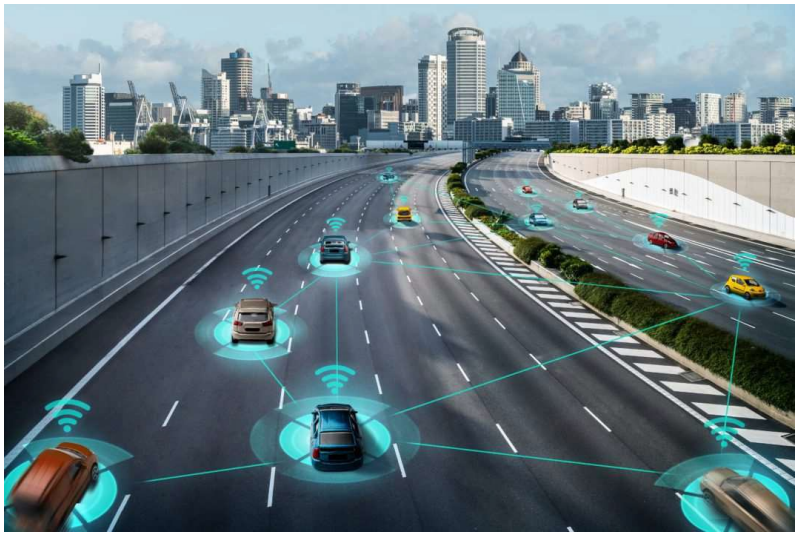
Şekil 3.2. Bir dakika içerisinde sosyal medya araçlarında gelen veriler (Web Kaynağı 3.2)

Büyük verinin diğer önemli bileşeni ise veri çeşitliliğidir. Verilerin belirli bir yapısı yoktur. Şekil 3.3'te gösterildiği gibi Sensörden alınan verilerin toplandığı .log dosyaları, resim, ses metin .txt dosyaları, şirket veri tabanlarının .csv (excel) dosyaları, twitterdan alınan .json verileri gibi birçok farklı veri çeşitliliği vardır.

Şekil 3.3. Büyük veride kullanılan bazı veri çeşitleri (Web Kaynağı 3.3)

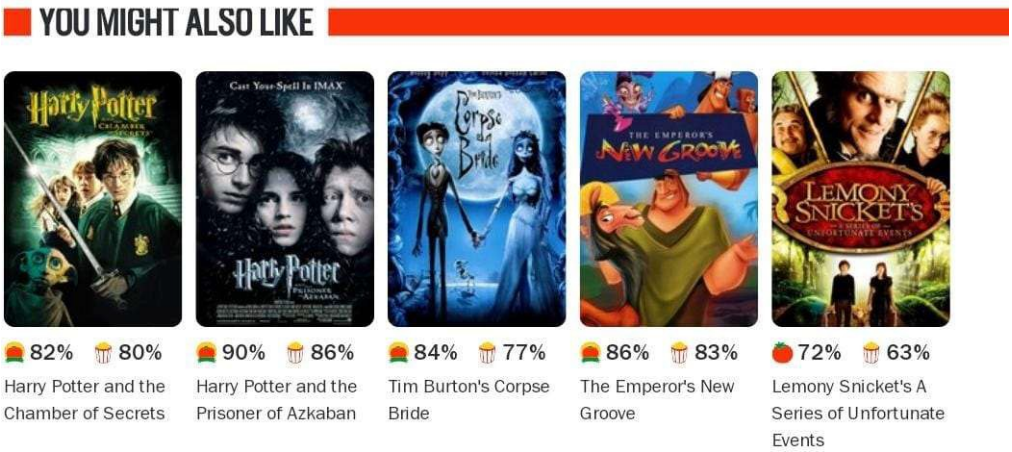
Büyük verinin dördüncü bileşeni ise verinin düzensiz, karmaşık ve kirli olmasıdır. Şekil 3.4'te gösterildiği gibi akan bir trafikte arabanın üzerinde bulunan sensörler vasıtasıyla araçların hızları alınmaktadır. Böylece araçların ortalama hızların göre, ileride karşılacakları trafik lambalarının

kırmızı veya yeşil yanma sürelerinin arttırılması ayarlanmaktadır. Ancak veriler analiz edilirken bir (1) tane aracın hızı için -10 km/saat olarak yanlış bir değer gelmektedir. Aracın hızının eksi (-) değer gelmesi oluşturulacak modelde yanlış bir sonuca neden olabilir. Bu durum verinin düzensiz ve karmaşık olmasına bir örnektir. Bu nedenle bu verilerin öncelikle doğru bir formata getirilmesi ya da veri setinden çıkarılması (temizlenmesi) gerekir.



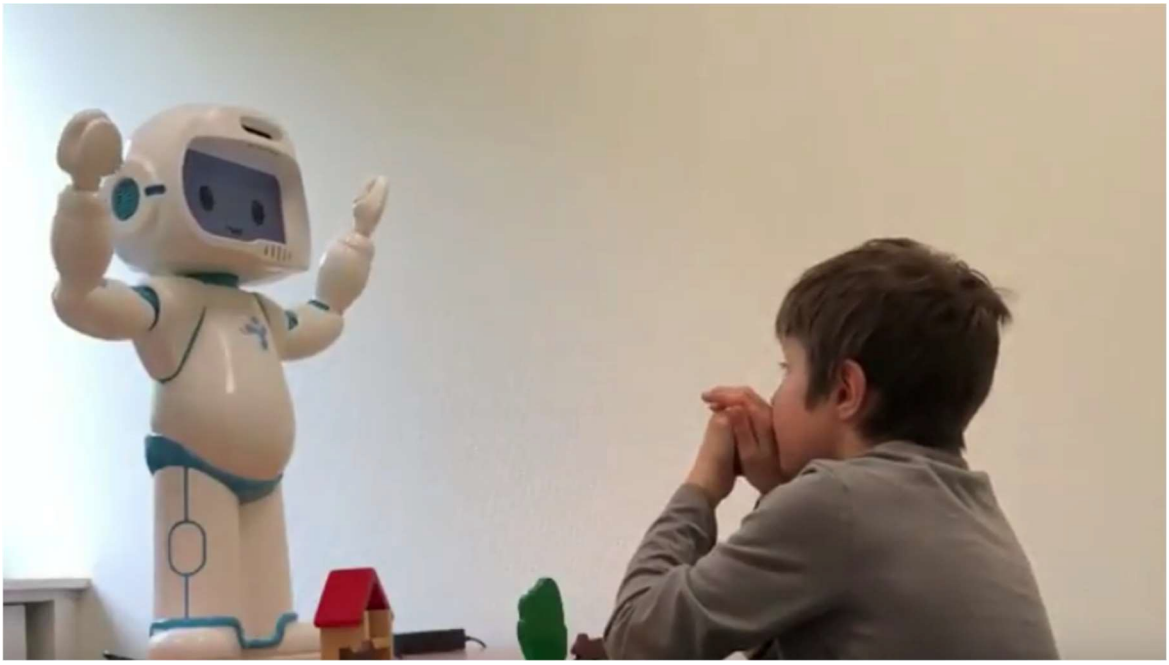
Şekil 3.4. Trafikte araçlardan sensörler vasıtasıyla hızların alınması görüntüsü (Web Kaynağı 3.4)

Büyük verinin son bileşeni verinin anlamlandırılmasıdır. Şekil 3.5'te gösterildiği gibi bir sinema uygulaması ile iki öğrenciden ilki farklı türde 10 tane film, diğeri ise yine ilk öğrenciye benzer 9 film almaktadır. Verinin anlamlandırılmasında 9 film alan öğrenciye benzer filmler alan öğrenciler incelenerek 10. (onuncu) film öğrenciye tavsiye olarak sunulmaktadır.



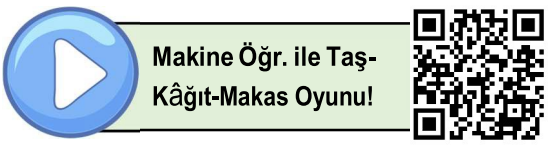
Şekil 3.5. Film tavsiye uygulaması (Web Kaynağı 3.5)

Büyük verileri anlamlandıran bireyler, kurumlar ve ülkeler kendilerini geliştirirken anlamlandıramayanlar ise zamanla teknolojinin gerisinde kalarak popülerliklerini ve sermayelerini kaybetmektedirler. Büyük veriyi değerlendirmek için makine öğrenme yöntemi sıklıkla kullanılmaktadır. Makine öğrenmesi akıllı cihazların verileri işleyerek, kendi kendine karar vererek bir problemi çözme sürecidir.



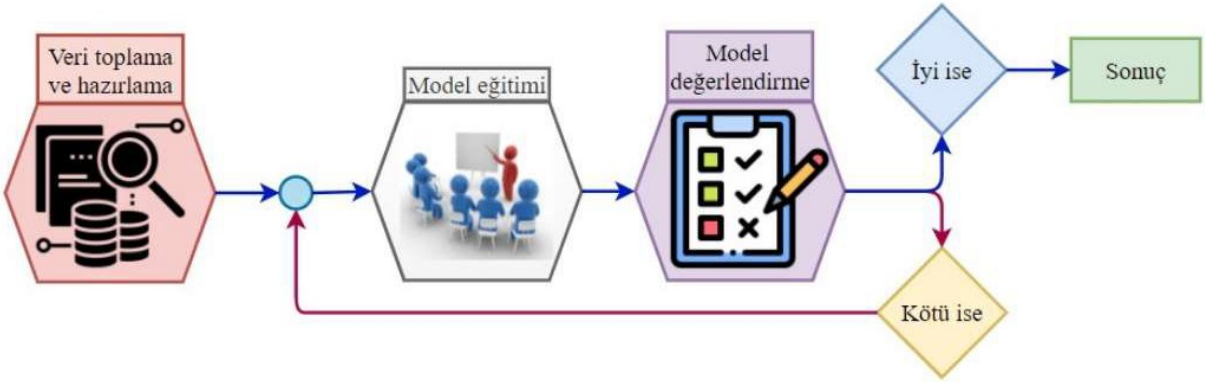
Şekil 3.6. İnsan robot etkileşiminde makine öğrenmesi (Web Kaynağı 3.6)

Makine öğrenmesi, bilgisayarın meydana gelen bir olay ile ilgili topladığı bilgi ve tecrübeleri öğrenebilmesi amacıyla matematiksel modellerin kullanılmasıdır. Makine öğrenmesinde temel amaç elde edilen veriler ile gelecekte oluşabilecek benzer olaylar hakkında kararlar verebilmek veya geçmişteki durumlar hakkında sonuç oluşturmaktır (Azure, 2021). Çok büyük miktarlardaki verilerin elle işlenip bir sonuca varılabilmesi oldukça zordur. Bu nedenle makine öğrenmesi algoritmaları kullanılarak bu işlem kolay ve kısa sürede gerçekleştirilebilir. Makine öğrenmesi, doğal dil işleme, nesne tanıma, arama motorları, robot hareket kontrolü, yüz tanıma gibi birçok uygulamada kullanılmaktadır (Bilgin, 2017).



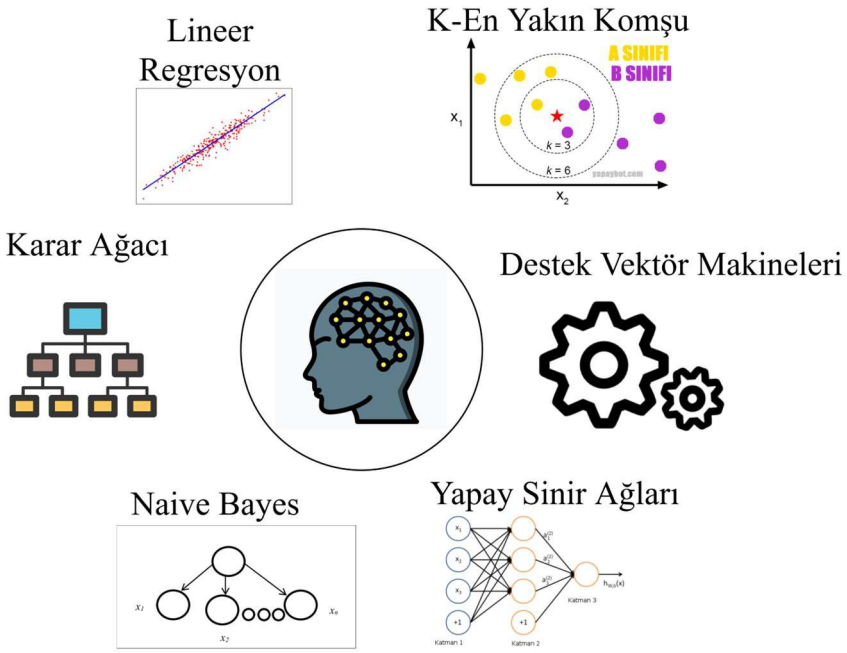
Şekil 3.7'de gösterildiği gibi makine öğrenmesi temel olarak dört aşamadan oluşmaktadır. İlk aşamada, akıllı cihazlardan veriler toplanarak işlenir. İşleme sürecinde uygun olmayan veriler veri setinden çıkarılarak veri bütünlüğü sağlanır. İkinci aşamada, veri setindeki veriler bir kısmı eğitim verisi diğer kısmı test verisi olarak ikiye ayrılır. Eğitim verileri makine öğrenmesindeki modelleri eğitmek için kullanılır. Üçüncü aşamada, modellerden elde edilen sonuçlar test verileri ile analiz

edilerek makine öğrenmesi modelinin doğruluğu test edilir. Son aşamada ise, test verilerinden elde edilen sonuçlar değerlendirilir (Çevik, 2020).



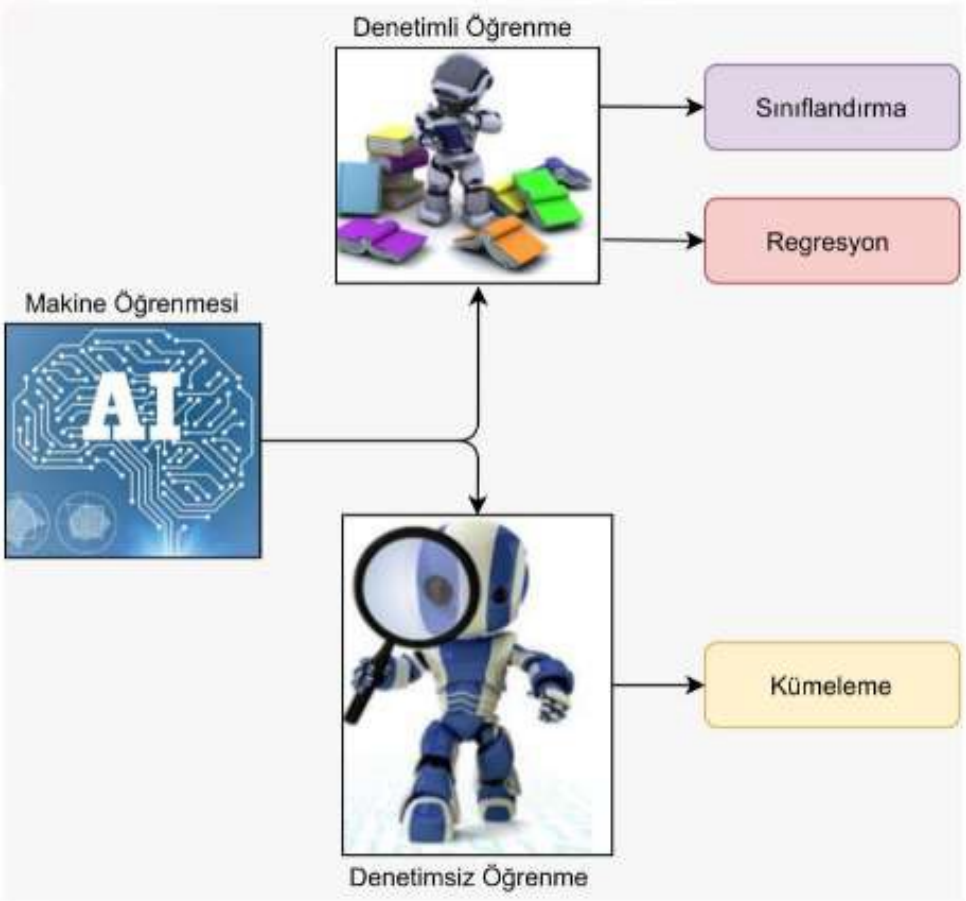
Şekil 3.7. Makine öğrenmesinde temel aşamalar

Makine öğrenmesi tekniklerinde sıklıkla naive-bayes algoritmaları, destek vektör makineleri, karar ağacı algoritmaları, k-en yakın komşu algoritmaları kullanılmaktadır (Şekil 3.8).



Şekil 3.8. Yaygın olarak kullanılan makine öğrenme algoritmaları

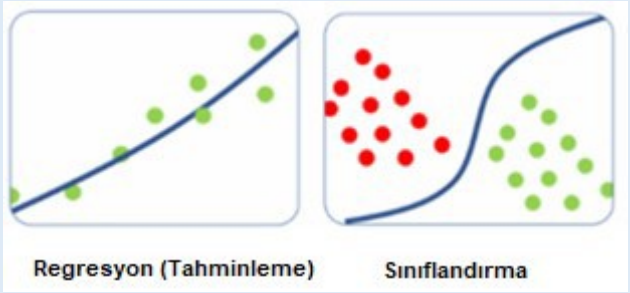
Bu algoritmalardan bir kısmı sınıflandırma uygulamalarında, bir kısmı da tahmin etme işlemlerinde kullanılır. Şekil 3.9'da gösterildiği gibi makine öğrenmesi algoritmaları denetimli ve denetimsiz öğrenme olmak üzere ikiye ayrılmaktadır (Şahinarslan, 2019).



Şekil 3.9. Makine öğrenme yöntemleri

Eğitmene Not

Eğitmen, öğrencilere regresyon ve sınıflandırma mantığını Şekil 3.10'da verilen görsele göre anlatır.



Şekil 3.10. Regresyon ve sınıflandırma görseli

1.2. Bayes Öğrenme Tekniği

Naive Bayes öğrenme tekniği temeli Bayes teoremine dayanır. Bayes Teoremi bir sonucun sebebini bulurken sonucun hangi olasılıkla hangi sebepten kaynaklandığını bulur. Ülkemizde en popüler alışveriş internet sitesinde akıllı cep telefonu satışını düşünelim. Online mağazada A ve B iki ayrı marka cep telefonu satılmaktadır. Edinilen tecrübe ve tutulan kayıtlardan cep telefonu ile ilgili bilgiler şu şekildedir:

1. Markalar; A ve B
2. Günlük satış miktarı; A marka 200, B marka 160 adet
3. İade edilen cep telefonu oranı; %5

Bir gün için online mağazada satılan cep telefonlarından A marka olanların iade edilme olasılığını hesaplayalım. Toplam 360 adet cep telefonu, 200 adedi A marka satılıyor. Bu durumda, günlük 18 adet cep telefon iade edilmektedir. İade edilen cep telefonların markalarına göre dağılımını eşit kabul edersek iade edilen cep telefonlarının 9 adeti A markasına aittir. O halde A marka cep telefonunun iade edilme olasılığı 9/200=%4,5'dur. Bu durumu özetleyen Bayes öğrenme formülü şöyledir:

$$P(\text{İade edilen Cep Tlf} \mid A) = \frac{P(A \mid \text{iade edilen cep tlf}) * P(\text{iade cep tlf oranı})}{P(\text{Satış oranı} \mid A)}$$

Formüle sayısal veriler uyarlandığında şu sonuç elde edilir:

$$P(\text{İade edilen Cep Tlf} \mid A) = \frac{0{,}5 * 0{,}05}{0{,}555} = 0{,}045(\%4{,}5)$$

Dünya'dan Haberler

Hazır Gıdada Yapay Zekâ.

New York merkezli bir araştırma ve geliştirme şirketi olan Analytical Flavor Systems, yapay zekâdan yararlanarak, gıda üreten şirketlere ürünlerini geliştirme ve yeni yiyecek ve içecekler geliştirme konularında danışmanlık yapıyor. Şirketin Gastrograph adındaki yapay zekâ platformu, örneğin herhangi bir yerel zevke hitap edecek bir içeceğin tadı, kokusu ve dokusunu tahmin edebiliyor. Dünyanın dört bir yanında 10 yıl boyunca tat-lezzet testleri yapan Jason Cohen "Bunun için çok çalıştık." diyor.

Cohen'e bağlı 50 "tadıcı" her gün iki ya da üç kez farklı hazır gıdaların tadına bakıyor. Covid-19 salgınından önce ekip her hafta farklı ülkelere seyahat ederek bölgesel lezzet tercihlerini de test ediyordu.

Eski bir çay eksperi olan Cohen "İnsanların bir şeyden ne tat aldıklarından dahaönemlisi ne tat aldıkları konusundaki algılarıdır." diyor ve algının çok kolay manipüle edilen bir şey olduğunu söylüyor. "Örneğin süte milyonda bir oranında vanilya eklersek, vanilyanın tadını alamazsınız, fakat sütün daha yoğun ve güzel olduğunu düşünürsünüz." diye açıklıyor. Yapay zekâ yazılımları iyi bir ürünün tadının tam olarak nasıl olması gerektiğini öğrenene kadar, o ürünün tadının nasıl olması gerektiği, eksperlerin yaptığı tadım sonuçları ve yerel lezzet tercihlerine dayanan yüzlerce olasılığı değerlendiriyor (Web Kaynağı 3.7).

2. TASARLA

Öğrenciler, ilk olarak bayes öğrenme ile araç satış değerlendirme problemini anlar. Şekil 3.11'de gösterilen aracın özelliklerine göre satış durumunu modeller.

Şekil 3.11. Araç teknik özellikleri

Öğrenciler Çizelge 3.1'de verilen Bayes öğrenme ile araç satış için giriş çıkış parametresini belirler.

Çizelge 3.1. Bayes öğrenme için giriş ve çıkış parametreleri

Veri Sayısı	GİRİŞ PARAMETRELERİ						ÇIKIŞ PARAMETRESİ
	Fiyat	Onarım	Kapı	Kişi	Bagaj Hacmi	Güvenlik	Satış
1	çok yüksek	çok yüksek	2	2	küçük	düşük	zor
2	çok yüksek	çok yüksek	2	2	küçük	orta	zor
3	çok yüksek	çok yüksek	2	2	küçük	yüksek	zor
4	çok yüksek	çok yüksek	2	2	orta	düşük	zor
5	çok yüksek	çok yüksek	2	2	orta	orta	zor
6	çok yüksek	çok yüksek	2	2	orta	yüksek	zor
7	çok yüksek	çok yüksek	2	2	büyük	düşük	zor
8	çok yüksek	çok yüksek	2	2	büyük	orta	zor
9	çok yüksek	çok yüksek	2	2	büyük	yüksek	zor
10	çok yüksek	çok yüksek	2	4	küçük	düşük	zor
11	çok yüksek	çok yüksek	2	4	küçük	orta	zor
12	çok yüksek	çok yüksek	2	4	küçük	yüksek	zor
13	çok yüksek	çok yüksek	2	4	orta	düşük	zor
14	çok yüksek	çok yüksek	2	4	orta	orta	zor
15	çok yüksek	çok yüksek	2	4	orta	yüksek	zor
16	çok yüksek	çok yüksek	2	4	büyük	düşük	zor
17	çok yüksek	çok yüksek	2	4	büyük	orta	Zor
...	...	...	...	...	...	...	...
1729	düşük	düşük	5 den fazla	4 den fazla	büyük	yüksek	çok kolay

Bu bölümde araçlara ait 1729 adet veri setinde (Dua, D. and Graff, C., 2019) fiyat, onarım, kapı, kişi sayısı, bagaj hacmi ve güvenlik giriş parametrelerine göre aracın satış durumunun makine öğrenmesi ile tahminlenmesi amaçlanmıştır.

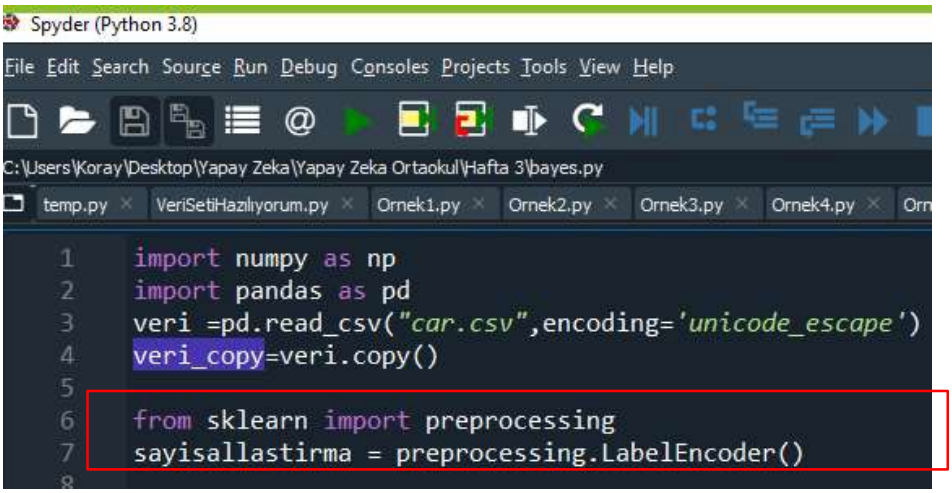
VERİ SETİ İÇİN İNDİRME LİNKİ

https://github.com/deneyapyz/ortaokul/blob/main/Hafta3/car.csv

3. HAREKETE GEÇ

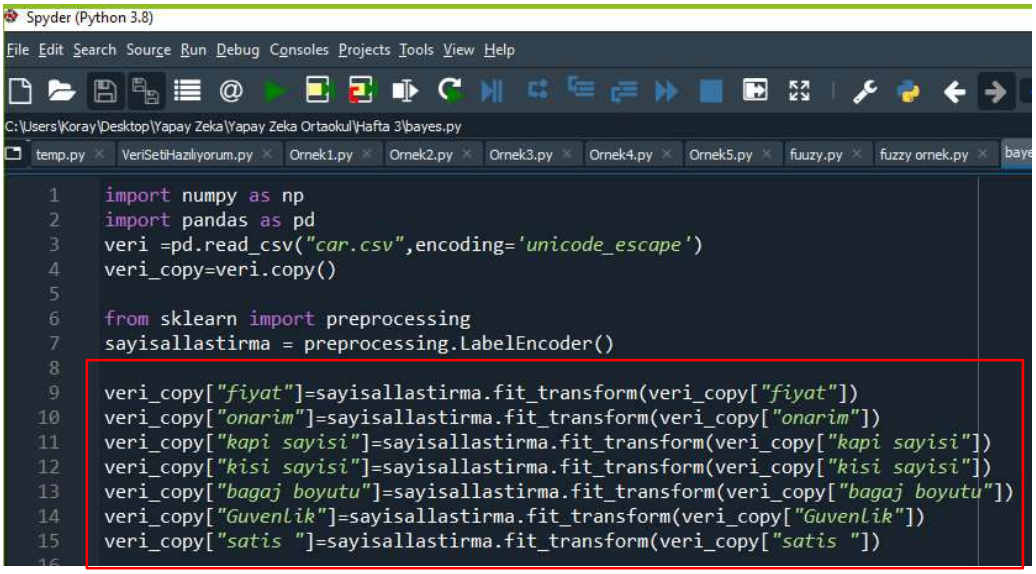
Öğrenci, verilen car.csv isimli veri seti dosyasını kullanarak bir Bayes öğrenme tahminleme işlemi yapar. Tasarım aşamasında fiyat, onarım, kapı, kişi sayısı, bagaj hacmi ve güvenlik parametrelerine göre aracın satış durumunu tahmin etmeye çalışır. Şekil 3.12'de gösterildiği gibi "car.csv" dosyasını yüklemek için gerekli kütüphane, komutlar ve veri_copy fonksiyonunu yazar (İlgili kod Github platformunda Hafta3 klasörü altında H3_bayes_arac-satisi.py dosyası ile sunulmuş durumdadır.).

Öğrenciler Şekil 3.13'de görüldüğü gibi veri setinde yer alan giriş ve çıkış parametrelerinin metinsel değerlerini sayısallaştırmak için Sklearn kütüphanesinde yer alan "ön işleme komutunu (preprocessing)" çağırarak **"preprocessing.LabelEncoder()"** fonksiyonunu çalıştırır. Böylece öğrenciler, "**sayisallastirma**" isimli değişken oluşturmuş olur.



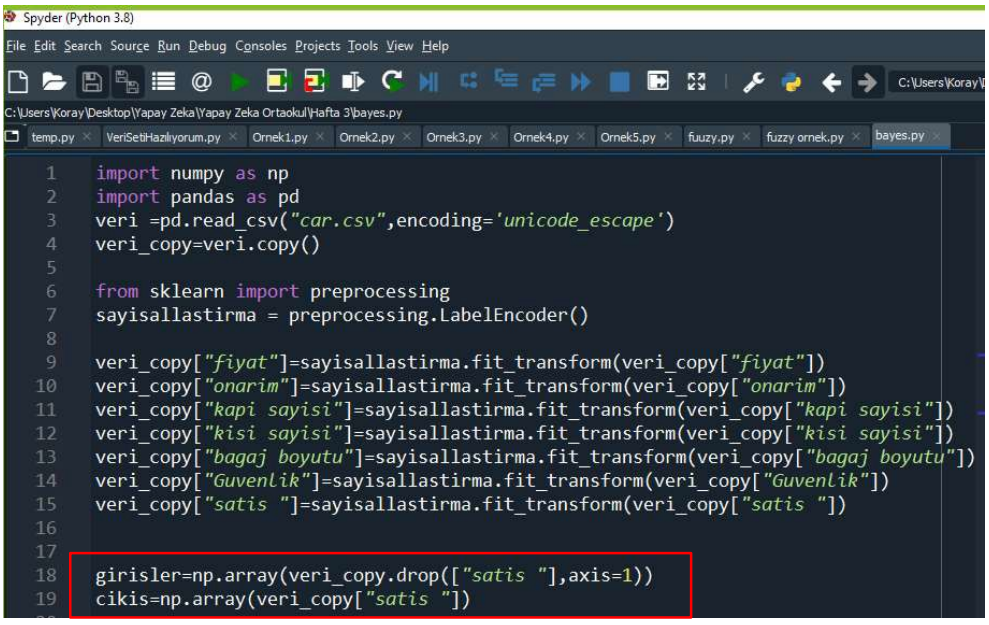
Şekil 3.13. Sayısallaştırmak için kütüphane ve fonksiyon çağırımı kod ekranı

Öğrenciler veri setinde yer alan fiyat, onarim, kapi sayisi, kişi sayisi, bagaj boyutu, güvenlik ve satis parametrelerinin değerleri olan iyi, normal, kolay, zor, çok kolay gibi metinsel ifadelerini bayes algoritması ile eğitim yapmadan önce Sklearn kütüphanesinde yer alan "**ön işleme (preprocessing)"** yöntemini kullanarak "0", "1", "2", "3" gibi sayısal değerlere dönüştürür. Öğrenciler Şekil 3.14'de görüldüğü gibi **sayisallastirma.fit_transform (veri_copy["giriş/çıkış parametresi"])** komutunu kullanarak verileri sayısallaştırır. Örneğin fiyat değeri "düşük, orta, yüksek" ve "çok yüksek" metinsel ifadeleri "0","1","2" ve "3" şeklinde sayısallaştırır.



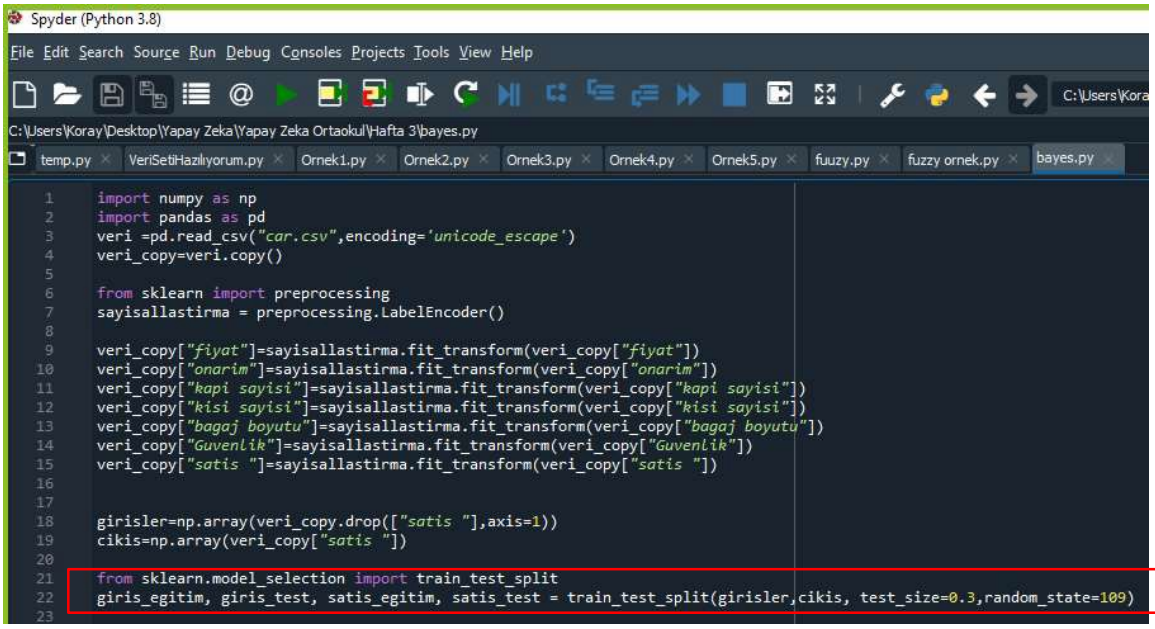
Şekil 3.14. Veri setindeki metin değerlerinin sayısallaştırılması kod ekranı

Öğrenciler Şekil 3.15'te gösterilen ilk komut satırında numpy kütüphanesindeki dizi (array) özelliğini kullanarak veri çerçevesi (data frame) formatındaki "car.csv" dosyasında son parametre olan **"satis"** parametresi hariç geriye kalan tüm parametrelerini **"veri_copy.drop"** komutu ile giriş parametresi olarak ayarlar. Burada yer alan "axis=1" ifadesi sütun yönünde silme işlemi yapılacağını belirtmektedir.



Şekil 3.15. Veri setindeki metin değerlerinin sayısallaştırılması kod ekranı

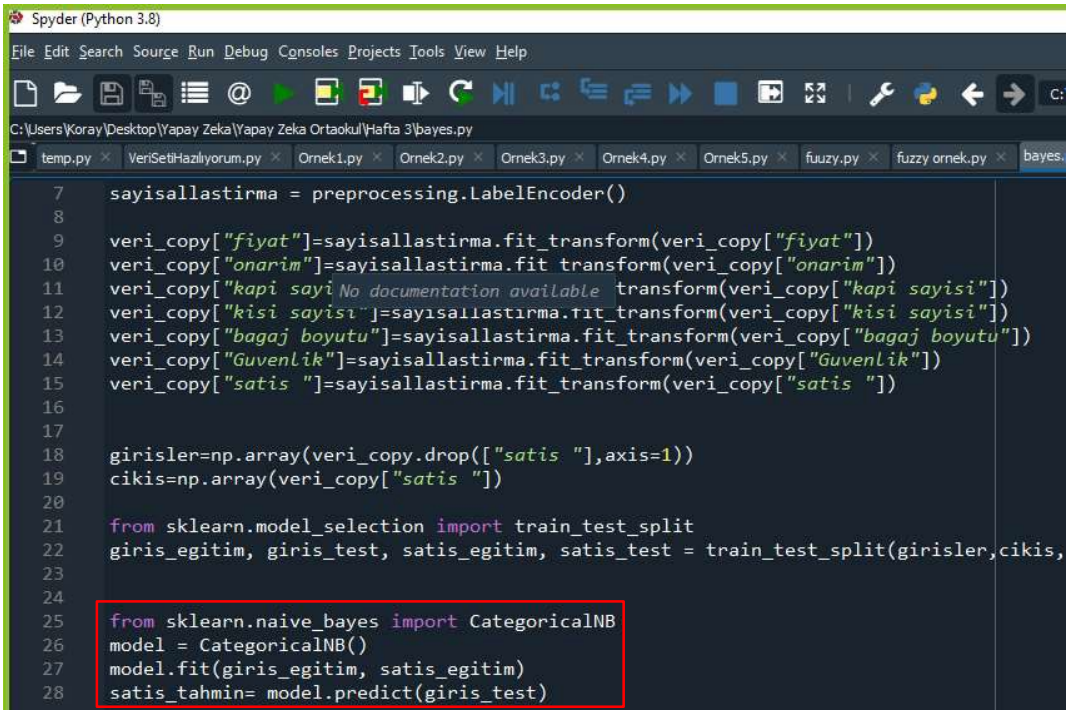
Öğrenciler ilk satırda, veri seti üzerindeki düzenlemeleri tamamladıktan sonra veri setini eğitim ve test olarak ayırabilmek için **"sklearn.model_selection"** kütüphanesinde yer alan **"train_test_split"** fonksiyonunu çağırır. İkinci satırda ise, veri setindeki eğitim ve test verilerini %70 eğitim, %30 test olacak şekilde **"test_size=0,3"** komutu kullanarak rastgele ayırır. Burada "**random_state=109"** ifadesi her eğitim tekrarında alınacak verilerin aynı kalmasını sağlar.



Şekil 3.16. Veri setinin eğitim ve test olarak rastgelen ayrılması için kod ekranı

4. YÜRÜT

Öğrenciler ilk satırda aracın özelliklerine göre satış durumu için oluşturmuş olduğu bayes öğrenme algoritmasını kullanabilmek için **"sklearn.naive_bayes"** kütüphanesinde yer alan **"CategoricalNB"** özellik fonksiyonunu çağırır. İkinci satırda **CategoricalNB()** fonksiyonu "**model**" isimli bir değişkene aktarılır. Üçüncü satırda, **"model.fit"** komutunu yazarak giriş eğitim verilerine göre çıkış eğitim verileri için Bayes algoritması ile eğitim gerçekleştirilir. Son kod satırında ise, Bayes algoritması ile eğitim işlemi sonunda giriş test verilerine göre satış tahmini gerçekleştirilir.



Şekil 3.17. Bayes algoritmasında eğitim ve tahminleme işlemleri için kod ekranı

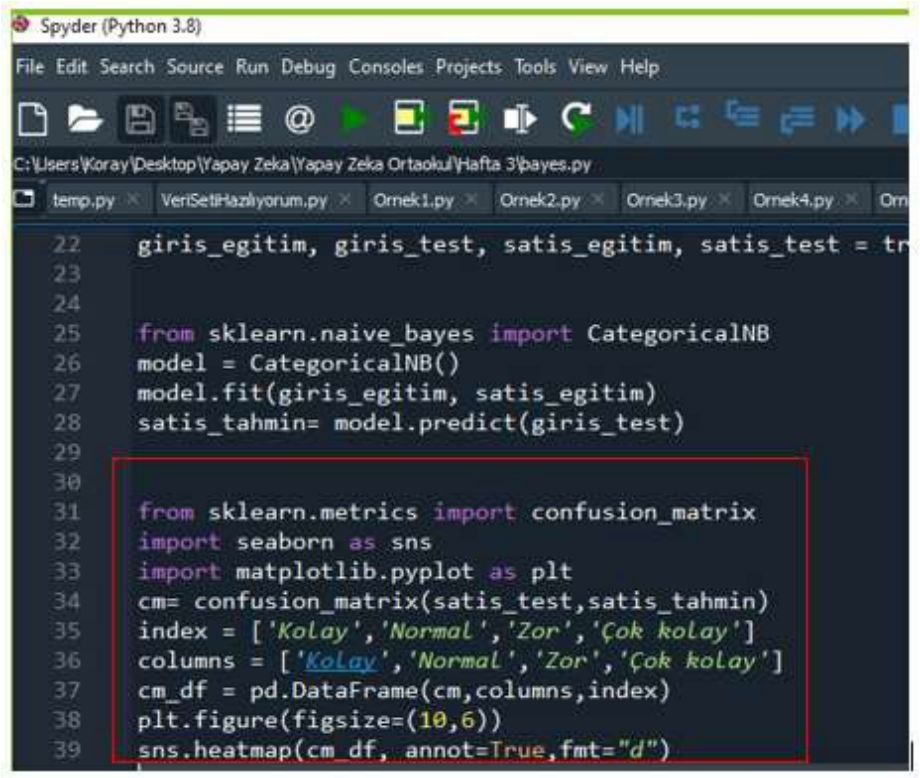
Eğitmene Not

Eğitmen, Makine Öğrenmesi algoritmalarında eğitim ve test süreçlerinin her zaman için uygulandığını ve bu süreçlerin gerekliliğini yeri geldikçe vurgular; öğrencilerin bu yönde farkındalığını geliştirir.

5. KARAR VER

5.1. Tahmin-Test Sonuçlarını Karşılaştırma

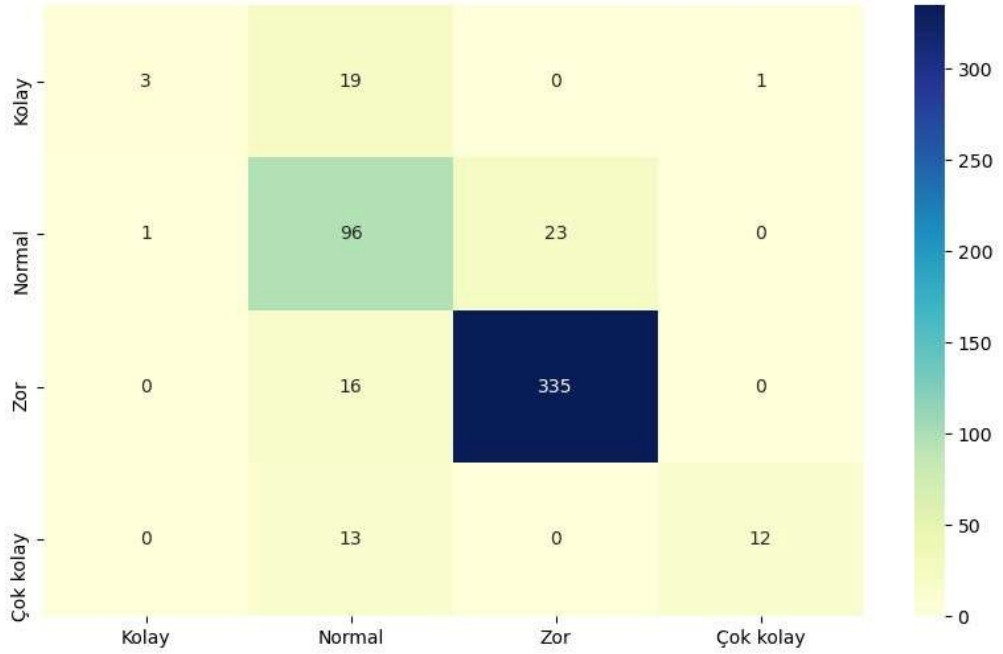
Bu aşamada, önceki aşamadaki uygulamaya eklemeler devam etmektedir. Öğrenciler, Bayes öğrenme modeli ile eğitilen araç satış durumu tahminlemesine ait hata matrisini (confusion matrix) kullanarak modelin başarısını ölçer. Şekil 3.18'de gösterildiği gibi 31 numaralı komut satırında, sklearn.metrics kütüphanesinden confusion_matrix özelliği yüklenir. 32 numaralı satırda ise, hata matrisini görüntüleyebilmek için gerekli olan seaborn kütüphanesini yükler. 33 numaralı satırda ise, hata matrisindeki grafikleri çizebilmek için **matplotlib** kütüphanesi içerisinde yer alan **pyplot** fonksiyonunu yükler. 34 numaralı kod satırında ise, cm değişkeni ile oluşturulan hata matrisinin satır (satis_test) ve sütunlarını (satis_tahmin) oluşturur. 35 ve 36 numaralı kod satırlarında ise index ve colums değişkenlerini kullanarak hata matrisinde yer alacak olan metinleri belirler. 37 numaralı kod satırında **cm_df** değişkeni ile satis_test ve satis_tahmin değerlerini veri çerçevesine (DataFrame) aktarılmasını sağlar. 38 numaralı kod satırında ise **"plt.figure"** komutu ile 10x6 cm çerçeve boyutunda boş bir çizim ekranı açar. 39 numaralı kod satırında ise **"sns.heatmap** komutu ile oluşturulan veri çerçevesini renkli olarak çizer. Burada annot=True ile sayısal değerler gösterilirken, fmt="d" ile sayısal değerler tam sayı olarak gösterilir.



Şekil 3.18. Bayes algoritmasında eğitim ve tahminleme işlemleri için kod ekranı

5.2. Hata Matrisini Değerlendirme

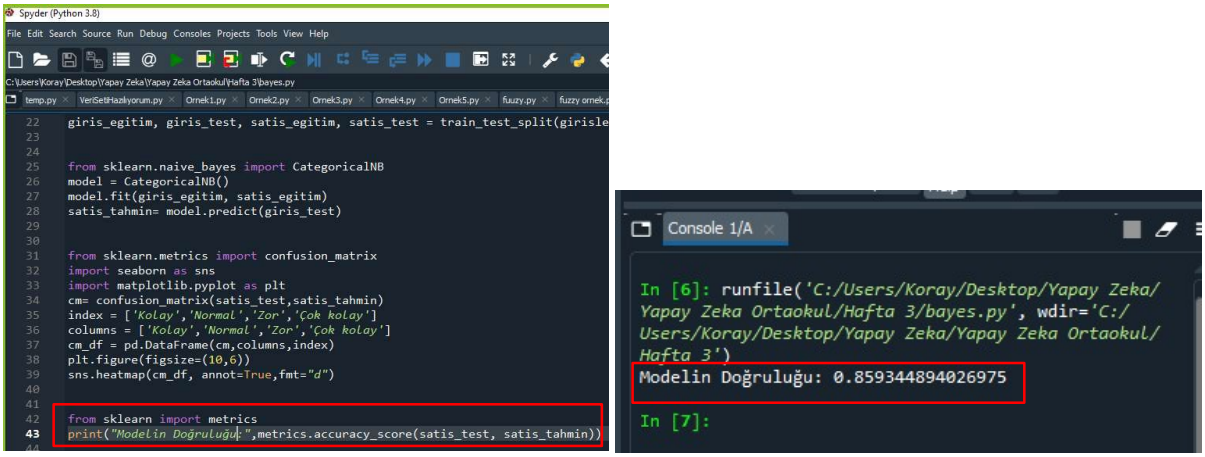
Öğrenciler, araç satış durumuna ait toplam 1729 adet veri setinde (Dua ve Graff, 2019) kayıt verinin %70'ini (1210) eğitim, %30'unu (519) test eğitim seti olarak ayırmıştı. Bayes modeli 1210 kayıt ile eğitmişti. Geriye kalan 519 kayıt (giris_test) ise modeli tahmin etmek için kullanıldı (satis_tahmin). Öğrenciler, Şekil 3.19'da gösterildiği gibi 519 test kaydına ait gerçek sonuçlar (satis_test) ile tahmin sonuçlarını karşılaştırdı ve hata matrisi elde etti. Böylece öğrenciler, matristeki sayılar toplamının test verisi olan 519'a eşit olduğunu görmüştür.



Şekil 3.19. Hata matrisi

Öğrenciler, hata matrisini incelediğinde 519 adet test kaydında araç satış durumuna ait 4 tane kolay test verisinden 3 adetin doğru (kolay), 1 adetin ise yanlış (normal) tahmin edildiğini görmüştür. 144 adet normal test verisinden 96 adetini doğru (normal), 19 adetini yanlış (kolay), 16 adetini yanlış (zor), 13 adetini yanlış (çok kolay) olmak üzere toplam 48 adetin yanlış tahminlendiğini görür. 358 adet zor test verisinden 335 adetin doğru (zor) ve 23 adetin yanlış (normal) tahminlendiğini görür. Son olarak 13 adet çok kolay test verisinden 12 adetin doğru (çok kolay) ve 1 adetin yanlış (kolay) olarak tahminlediğini görür.

Öğrenciler Şekil 3.20'de gösterildiği gibi 42 numaralı kod satırında bayes öğrenme modeli ile eğitilen modelin doğruluğunu değerlendirmek için "**sklearn"** kütüphanesinden **"metrics"** fonksiyon özelliğini yükler. 43 numaralı kod satırında "**print"** komutunu kullanarak **"satis_test"** veri setindeki veriler ile **"satis_tahmin"** işlemi ile modelin doğruluğunun konsol ekranında %85,9 başarım oranında tahminlendiğini görür.



Şekil 3.20. Bayes öğrenme ile araç satış durumunu için a) model doğruluk belirleme kod ekranı b) model doğruluk sonucu

Düşün, tartış…

Makine Öğrenmesi'nin çalışma şekli insanların öğrenme şekline benzer midir? Tartışalım…

6. İLAVE ETKİNLİK

Şekil 3.21'de gösterildiği gibi hayvanat bahçesindeki verilen bilgilere göre hayvanın cinsini tahminleyen Bayes öğrenme modeli kurunuz (İlgili kod Github platformu Hafta 3 klasörü altında H3_bayes_hayvanat-bahcesi.py adlı dosya ile sunulmuş durumdadır).

Şekil 3.21. Hayvanat bahçesinden görüntü

VERİ SETİ İÇİN İNDİRME LİNKİ

https://github.com/deneyapyz/ortaokul/blob/main/Hafta3/hayvanatbahcesi.csv

Eğitmene Not

Eğitmen, öğrencilerden ilgili indirme linkini kullanmak suretiyle, orijinal veri setinin (https://archive.ics.uci.edu/ml/datasets/zoo) Türkçe'ye dönüştürülmüş versiyonunu indirmelerini ister:

sac varligi	Tuy varligi	yumurta	sut	havada yasami	suda yasami	yirtici	dis varligi	omurga	nefes alimi	zehirli mi	yuzgecler	bacak sayisi	kuyruk	yerli	kedi boyu mu

Eğitmen, çizelgede verilen giriş değişkenlerinin (bacak sayısı hariç) hayvanlarda var ise "1" yok ise "0" olduğunu öğrencilere açıklar. Bacak sayısı ise 0,2,4,6,8 şeklinde olduğunu belirtir.

Eğitmen öğrencilere, şu çizelgede verilen çıkış **"sınıf"** parametresindeki hayvan türlerine ait sınıfları açıklar:

1	2	3	4	5	6	7
Yer domuzu, antilop, ayı, domuz, bufalo, buzağı, tüylü kemirgen, çita, geyik, yunus, fil, meyve-yarasa, zürafa, kız, keçi, goril, hamster, tavşan, leopar, aslan, vaşak, vizon, köstebek, firavun faresi, keseli sıçan, ornitorenk, sansar, midilli, liman-yunusu, puma, kedi, rakun, ren geyiği, fok, fok balığı, sincap, vampir, tarla faresi, ufak kanguru, kurt	Tavuk, karga, güvercin, Ördek, flamingo, martı, Şahin, Kivi kuşu, tarlakuşu, devekuşu, muhabbet kuşu, penguen, Sülün, Amerika devekuşu, Deniz süpürücüsü, Korsanmartı, serçe, kuğu, akbaba, çalıkuşu	çukur engerek, deniz yılanı, yavaş solucan, kaplumbağa, tuatara	bas, sazan, yayın balığı, Şap, köpek balığı, mezgit balığı, ringa balığı, Turna, Pirana, Denizatı, Dilbalığı, Vatoz, Ton balığı	kurbağa, semender, kara kurbağası	pire, tatarcık, bal arısı, karasinek, uğur böceği, güve, termit, yaban arısı	deniz tarağı, yengeç, kerevit, ıstakoz, ahtapot, akrep, deniz arısı, sümüklü böcek, deniz yıldızı, solucan

PYTHON KODLARI:

```
import numpy as np
import pandas as pd
veri =pd.read_csv("hayvanatbahcesi.csv",encoding='unicode_escape')

girisler=np.array(veri.drop(["sinifi"],axis=1))
cikis=np.array(veri["sinifi"])

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(girisler,cikis,
test_size=0.35,random_state=109)

from sklearn.naive_bayes import CategoricalNB

gnb = CategoricalNB()
gnb.fit(X_train, y_train)
y_pred = gnb.predict(X_test)

from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
cm= confusion_matrix(y_test,y_pred)
index = ['1','2','3','4','5','6','7']
columns =  ['1','2','3','4','5','6','7']
cm_df = pd.DataFrame(cm,columns,index)
plt.figure(figsize=(10,6))
sns.heatmap(cm_df, annot=True,fmt="d")

from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Eğitmene Not

Eğitmen, öğrencilerin ilgisini arttırmak ve yeni öğrenilen bilgileri önceden öğrenilen bilgiler ile karşılaştırmalarını sağlamak için aşağıda maddeler halinde verilen soruları tartışır.

- İlgili haftada ele alınan yapay zekâ uygulamalarına uygun örnek veri setleri neler olabilir?
- İlgili haftada ele alınan veri setlerinin yapay zekâ teknik ve kütüphanelerinde ne derecede başarım etkisi olmuştur?
- İlgili haftada öğretilen yapay zekâ tekniği ile uyumlu ve uyumsuz olabilecek açık erişimli veri seti çeşitleri sunan (kaagle, github vs.) internet sitelerinden elde edilecek farklı veri çeşitleri ile yapay zekâ tekniği uyumluluğu nasıldır?

Kaynakça

Bilgin, M. (2017). Gerçek Veri Setlerinde Klasik Makine Öğrenmesi Yöntemlerinin Performans Analizi. Breast, 2(9), 683.

Çevik, K. K., & Kayakuş, M. (2020). Bilişim Teknolojileri Departmanında Kullanıcıların Taleplerine Cevap Verme Süresinin Makine Öğrenmesi İle Tahmin Edilmesi. Mühendislik Bilimleri Ve Tasarım Dergisi, 8(3), 728-739.

Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Şahinarslan, F. V. (2019). Makine Öğrenmesi Algoritmaları İle Nüfus Tahmini: Türkiye Örneği (Doctoral dissertation, Sosyal Bilimler Enstitüsü).

Web Kaynağı 3.1: https://blogs.sap.com/2015/01/20/why-airlines-need-to-keep-planes-in-the-air/

Web Kaynağı 3.2: https://www.domo.com/blog/data-never-sleeps-3-0/

Web Kaynağı 3.3: https://erdicenger.medium.com/big-data-nedir-fcd4a4a09d1

Web Kaynağı 3.4: https://isibilenesor.files.wordpress.com/2021/10/news-aus-1.jpg?w=848

Web Kaynağı 3.5: https://www.business2community.com/small-business/value-product-recommendation-marketing-small-businesses-0985412?hcb=1

Web Kaynağı 3.6: https://bilimgenc.tubitak.gov.tr/makale/robotlar-otizmli-cocuklarin-sosyal-becerilerini-gelistirmeye-yardimci-olabilir

Web Kaynağı 3.7: https://www.bbc.com/turkce/haberler-dunya-54075570

4. Hafta: Karar Ağaçları ile Tutarlı Kararlar

Ön Bilgi:

- Karar ağaçları temelleri, Karar ağaçları tekniği, Karar ağaçları tekniğinde çözüm modelleri, Karar ağaçları ile COVID-19 hastalığı teşhis uygulaması.
- Python kodlamada döngüler, fonksiyonlar ve kütüphane yapılarının kullanımı (Bu konuda Python anlatım videoları: 10-11. Döngüler, 13. Fonksiyonlar, 14-16. Python kütüphaneleri serisi yardımcı araç olarak kullanılabilecektir).

Haftanın Kazanımları:

- Öğrenciler, karar ağaçları algoritmasının temel yapısını kavrar.
- Öğrenciler, verileri karar ağaçları ile gerçek hayat problemlerine uygular.
- Öğrenciler karar ağaçları algoritmaları ile tahmin, sınıflandırma yöntemlerini bilir.
- Öğrenciler, Karar ağaçları üzerindeki koşul ifadeleri için 'düğüm' ve 'kök düğüm', Düğümler arası bağlantı için 'kenar' ve koşul belirtmeyen ifadeler için ise 'Yaprak' kavramlarını bilir.
- Öğrenciler Python programlama dilini kullanarak aracın özelliklerine göre pirinç yaprağı hastalıklarının teşhisi için program kodunu oluşturur.

Haftanın Amacı:

Bu haftanın amacı, "karar ağaçları algoritması" kavramının tüm öğrenciler tarafından doğru bir şekilde anlaşılmasını sağlamaktır. Haftanın bir diğer amacı, karar ağaçları algoritması kullanılarak farklı örnekler ile öğrencilerin ilgisini çekerek etkinliklere yönelik ilgilerini artırmaktır. Ayrıca, öğrencilere Python ile karar ağaçları algoritmaları kullanarak örnek modelleme ve çözüm üretme yetenekleri kazandırılacaktır.

Kullanılacak Malzemeler:

Bilgisayar, kâğıt, kalem, Python kod editörü, örnekler için çıktı görselleri

Haftanın İşlenişi:

Algıla: Karar ağaçları tekniği kavramlarını öğrenir.

Tasarla: Karar ağaçları ile COVID-19 hastalığı teşhisi probleminin giriş ve çıkış parametrelerini tasarlar. Bu hafta için Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşim Tasarla aşaması ile başlatılabilmektedir.

Harekete Geç: Python ile veri setindeki görüntü değerlerini sayısallaştırıp eğitim ve test verilerini ayırarak model kurar.

Yürüt: Eğitmenler öğrenciler ile etkileşimli bir biçimde karar ağaçları modelini eğitir ve COVID-19 hastalığının teşhisi için tahmin durumunu belirler.

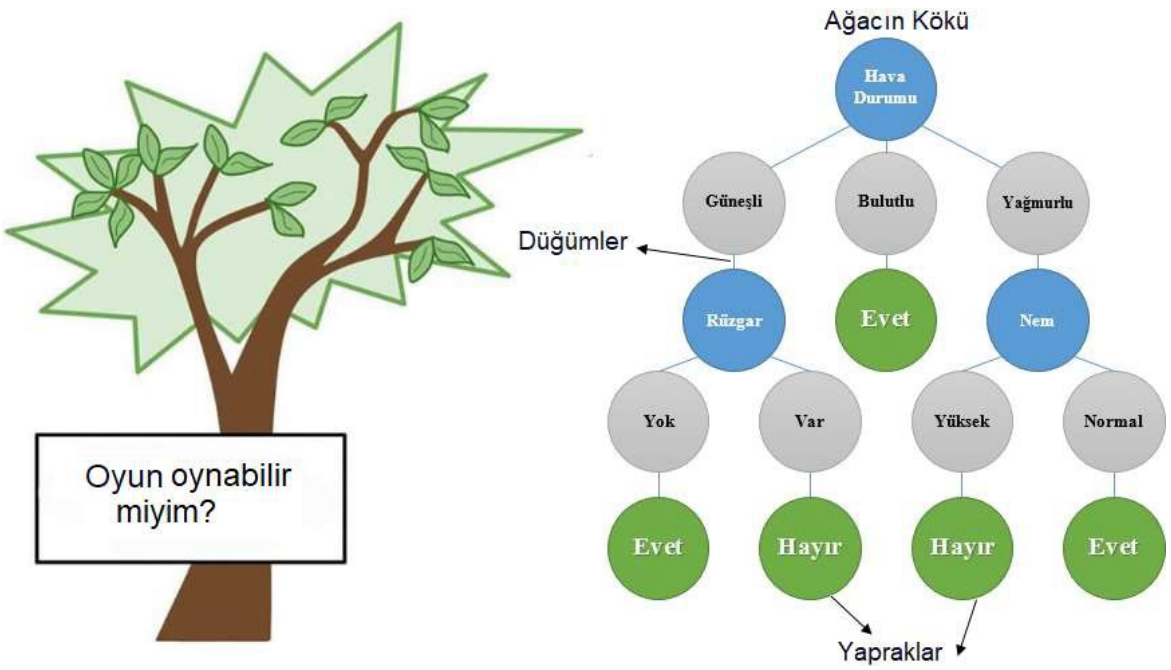
Karar Ver: Haftanın içeriğinin Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşimli kullanımı. Öğrenciler özellikle görsel veriler üzerindeki uygulamaların işleyişini inceler ve tartışır.

1. ALGILA

1.1. Karar Ağaçları Algoritması Temelleri

Karar ağaçları, sınıfları bilinen örnek veriden karar düğümleri (decision nodes) ve yaprak düğümleri (leaf nodes) oluşturarak ağaç şekilli bir karar akışı çıktısı veren yapay zekâ algoritmasıdır (SPSS, 1999). Karar ağaçları algoritması, veri setini bölüp küçülterek geliştirilen bir yöntemdir. Karar düğümleri bir veya birden fazla daldan meydana gelebilir. İlk düğüme kök düğüm (root node) denir. Karar ağacı algoritmaları hem metinsel hem de sayısal verilerden oluşabilir (Web Kaynağı 4.1).

Şekil 4.1'de gösterildiği gibi havanın durumuna göre öğrencilerin okulun bahçesinde oyun oynayıp oynayamayacağını belirlemek için karar ağaçları yapısı gösterilmiştir. Burada karar ağacının algoritmasının kökü, hava durumudur. Karar ağacının düğümleri ise güneş, bulut ve yağmurdur. Rüzgâr ve havanın nemi de koşul ifadeleridir. Karar ağacının yaprakları ise öğrencilerin okulun bahçesinde oyun oynayıp oynamayacağını belirtir.



Şekil 4.1. Karar ağaçlarının yapısı (Web Kaynağı 4.1)

Karar ağaçları avantaj ve dezavantajları şu şekilde sıralabilir:

Avantajlar

- Ağaç yapılarının yorumlanması ve görselleştirilmesi kolaydır.
- Hem metinsel hem de sayısal veriler analiz edilebilir.
- Karar ağaçları, giriş parametrelerine bağlı çarpanlardan oluşan denklem şeklinde değil, koşullu bir çıkış modeli verir. Bu nedenle modelin çalışması hızlıdır.
- Yüksek miktarda veriye ihtiyaç duymadan model eğitimi gerçekleştirilebilir.
- Birden fazla çıkış parametresine sahip problemleri çözebilir.

Dezavantajlar

- Verilerin analizi yapılırken otomatik oluşan büyük ağaç yapılarının aşırı karmaşık olmasından dolayı ağaç dallarının takibi zordur.
- Aşırı öğrenme (over fitting) yaşanabilir.

Eğitmene Not

Eğitmen, öğrencilere aşırı öğrenme (over fitting) kavramını şöyle bir örnek ile açıklar:

Algoritmanın eğitim verisi üzerinden veriyi ezberlemesine verilen tekniğin adıdır. Aşırı öğrenmede eğitim verilerinden elde edilen sonuç yüksek olurken, eğitim verisinden farklı bir veri modele verildiğinde hatalı sonuçlar üretir. Kısaca model kararlı bir yapıya sahip değildir (Web Kaynağı 4.2).

Dünya'dan Haberler

Mucit Yapay Zekâ

Surrey Üniversitesinde profesör olan Ryan Abbot tarafından 2019'da başlatılan The Artificial Inventor adlı proje, yapay zekâ ve yasanın yollarının kesişmesine odaklanıyordu. DABUS olarak adlandırılan bir sistem için projenin başlamasıyla birlikte 2 patent başvurusunda bulunulmuştu. Ayarlanabilir bir yemek kabı ve acil durum işaret ışığı olan icatların patentleri, mucit olarak DABUS'u gösteriyordu.

Dr. Stephen Thaler tarafından geliştirilen DABUS adlı sistem, sahip olduğu trilyonlarca ağ sayesinde büyük fikir ve icatlar öne sürebiliyordu. Fakat Amerika başta olmak üzere birçok ülke, yapay zekânın mucit olamayacağını söyleyerek patent tekliflerini geri çevirdi. Bunun üzerine davalar açan Thaler ve Abbott, son gelişmelerle birlikte bekledikleri sonuca yavaş yavaş ulaşacak gibi görünüyorlar.

Başlangıçta Avustralya da bu fikre karşı çıkıyordu. Fakat Avustralya Federal Mahkemesinden Jonathan Beach, bu durum karşısındaki tavrın değişmesine karar verdi. Beach'in vardığı karara göre DABUS, mucit olarak görülecek, fakat patent için aday olamayacak veya başvuramayacak. Bu noktada patentin sahibi, DABUS'un geliştiricisi Thaler olacak.

Beach, yaptığı açıklamada kendi görüşüne göre yapay zekâ sisteminin veya cihazın mucit olarak tanınabileceğini söylüyor: "İkimiz de yaratıldık ve yaratıyoruz. Bizim yarattıklarımız neden yaratamasın?" (Web Kaynağı (4.3).

t iliyor musunuz?

Karar Ağaçları ile karar verme aşamasında hangi durumların birbirini takip edeceğine Entropi adı verilen bilgi kazancı hesabıyla karar verilir. Bizi karara ulaştırabilen her durumdan hangisinin Entropi değeri yüksek çıkarsa ağaçla sıradaki düğüme o eklenir

1.2. Karar Ağacı Tekniği

Karar ağacı tekniğini kullanarak verinin sınıflanması, öğrenme ve sınıflama olmak üzere iki basamaklı bir işlemdir. Yaygın olarak bilinen karar ağacı algoritmaları şu şekildedir:

1.2.1. ID3 Karar Ağacı Algoritması

ID3, yapay zekâ alanında çalışmaya ilk başlayan öğrencilerin karar ağaçlarının temel çalışma şeklini kolayca öğrenebilecekleri ideal bir algoritmadır. ID3 karar ağaç algoritmasının C4.5 ve C5.0 isminde iki tane versiyonu sıklıkla kullanılmaktadır. ID3 karar ağacı algoritmasında her düğümden çıkan dallar ile karar ağacı oluşmaktadır. Ağaçtaki dalların sayısı algoritmada tahmin edilecek sınıf sayısına eşittir. Karar ağacı algoritmasında yapraktaki hata (error) oranına göre budama işlemi yapılır.

Eğitmene Not

Eğitmen, öğrencilere karar ağaçları ile ilgili aşağıda verilmiş olan örneği tartışarak çözer. Bir şirketin A, B, C ürünleri ile ilgili satış detayları Aşağıdaki Tablo'da verilmiştir. Bu ürünler ile aşağıda belirlenen özelliklerde ürünler ile aksesuar alma kararı da sette tanımlanmıştır. Tabloya uygun olarak basit bir karar ağaç yapısı oluşturunuz.

No	Tip	Ürün Sayısı	Ciro	Yurtiçi	Aksesuar
1	A	<=5	düşük	evet	hayır
2	A	<=5	düşük	evet	hayır
3	B	5…10	orta	evet	evet
4	A	<=5	orta	evet	hayır
5	A	<=5	yüksek	evet	hayır
6	B	>10	orta	evet	evet
7	A	<=5	orta	evet	evet
8	A	<=5	orta	evet	hayır
9	C	<=5	orta	evet	evet

1.2.2. C&RT Karar Ağacı Algoritması

Gini indeksi (dizini) veya Gini katsayısı, İtalyan istatistikçi Corrado Gini tarafından 1912'de geliştirilen istatistiksel bir ölçüdür. Gini'ye dayalı ikili bölme işlemine göre çalışan bir karar ağacı algoritmasıdır. Bu algoritmada en son veya uçta olmayan her bir düğümde iki adet dal vardır. Hem sınıflandırma hem de regresyon (sayısal sonuç) uygulamalarında kullanılır. Budama işlemi oluşturulan karar ağacı yapısına göre değişiklik gösterir.

1.2.3. CHAID Karar Ağacı Algoritması

Karar ağacı CHAID algoritması istatistik tabanlı olarak G. V. Kass tarafından 1980'de geliştirilmiştir. Sınıflandırma ve regresyon uygulamalarında tercih edilir. CHAID algoritması, bağımsız değişkenlerin birbirleriyle olan etkileşimini bulan bir tekniktir. CHAID algoritması dallanma kriterinde bağımlı değişken kategorik ise iki ya da daha çok grup arasında fark olup olmadığını tespit eden Ki-kare testine göre bölme işlemini gerçekleştirir.

1.2.4. SPRINT Karar Ağacı Algoritması

SPRINT algoritması 1996 yılında Shafer, Agrawal ve Mehta tarafından geliştirilip entropiye dayanmaktadır. SPRINT karar ağaçları algoritması büyük veri kümeleri için ideal bir algoritmadır. Ağaç yapısında en iyi dallanma için her bir değişkene ait özellikleri bir kez sıraya dizer ve karar ağaçı yapısı bu şekilde oluşur. Bu algoritmada her bir değişken için ayrı bir değişken listesi hazırlanır. Bölme işlemi tek bir özelliğin değerine göre saptanır.

1.2.5. SLIQ Karar Ağacı Algoritması

SLIQ karar ağacı algoritması 1996 yılında Agrawal, Mehta ve Rissanen tarafından geliştirilmiştir. Bu algoritma Gini tekniği ile nicel ve nitel veri tipleri kullanabilmektedir. Ayrıca verilerin sıralanması aşamasında en iyi dallara ayırma tekniğini uygulamaktadır. Bu algoritma hızlı ölçüm yapan bir sınıflandırıcıya ve hızlı ağaç budama algoritmasına sahiptir.

Düşün, tartış…

Yapay Zekâ'nın karar verme süreçlerinde biz insanları desteklemesi her zaman faydalı olur mu? Olası riskleri var mıdır? Tartışalım!

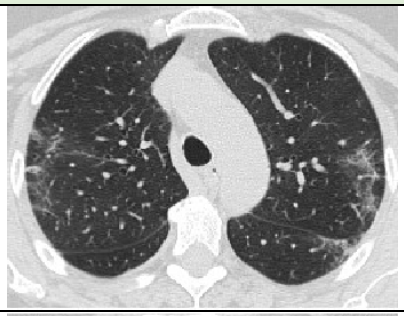
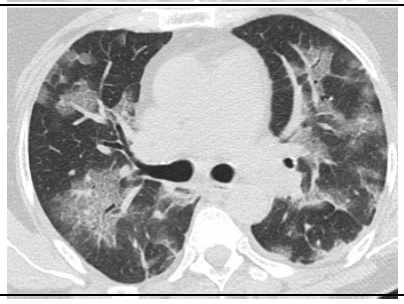
2. TASARLA

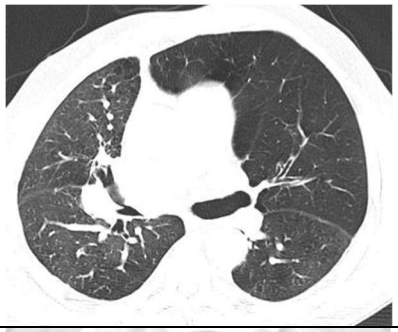
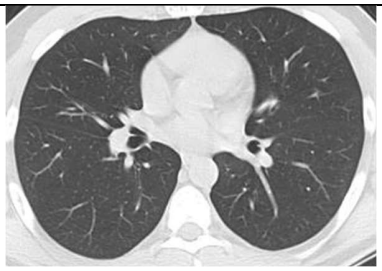
Öğrenciler, ilk olarak karar ağaçları ile Bilgisayar Tomografisi (BT) üzerinde COVID-19 hastalığının değerlendirilmesi problemini anlar. Şekil 4.2'de gösterilen BT görüntüsüne göre COVID-19 hastalığı olup olmadığını modeller.

Şekil 4.2. BT görüntüsü

Öğrenciler Çizelge 4.1'de verilen karar ağaçları algoritması ile COVID-19 hastalığı için giriş çıkış parametresini belirler.

Çizelge 4.1. Karar ağaçları algoritması için giriş ve çıkış parametreleri

Veri Sayısı	GİRİŞ PARAMETRESİ	ÇIKIŞ PARAMETRESİ
	BT Görüntüsü	HASTALIK
1		0 (Pozitif)
2		0 (Pozitif)
...	...	...

1252		0 (Pozitif)
1253		1 (Negatif)
...	...	...
2481		1 (Negatif)

Bu bölümde hastalara ait 2481 adet veri setinde (Kaggle, 2021) BT görüntüsü giriş parametrelerine göre hastanın COVID-19 olup olmaması durumunun karar ağaçları algoritması ile tahminlenmesi amaçlanmıştır.

Eğitmene Not

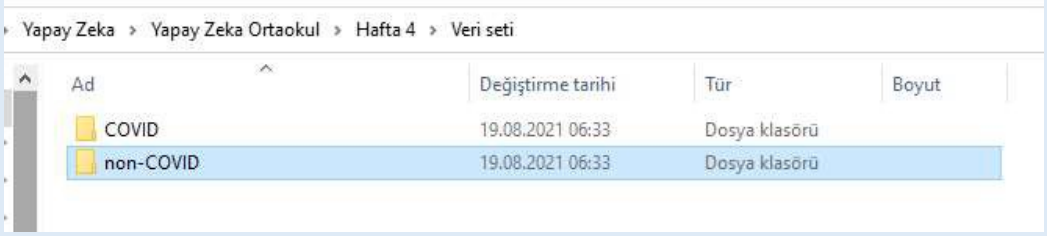
Eğitmen, öğrencilere Kaggle, Github gibi açık erişimli internet sitelerinin tüm dünyada veri seti olarak ortak kullanıma açılmış olduğunu açıklar. Bu uygulamada kullanılacak veri seti Kaggle ortamında yer aldığı gibi aynı zamanda Github platformuna da eklenmiştir. Eğitmen uygulamalarda bu tür platformlarla etkileşimi vurgular.

VERİ SETİ İÇİN İNDİRME LİNKİ

https://www.kaggle.com/plameneduardo/sarscov2-ctscan-dataset

Eğitmene Not

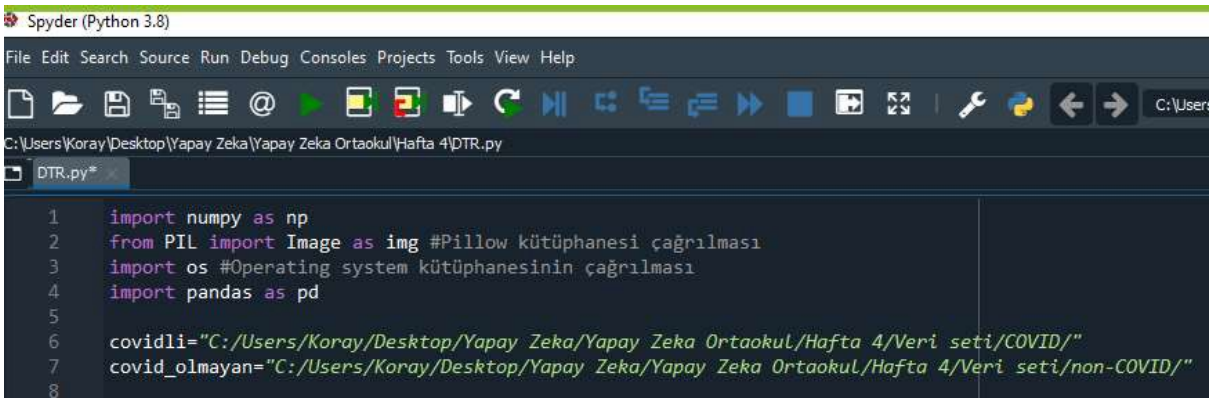
Eğitmen, Şekil 4.3'te gösterildiği gibi öğrencilere veri setini anlatırken veri seti indirme linkinden indirilen verilerin COVID ve non-COVID olarak ayrı bir şekilde indirildiğini anlatır.



Şekil 4.3. Veri seti klasörleri

3. HAREKETE GEÇ

Öğrenci ilgili veri seti dosyasını basit bir karar ağaçları algoritması ile COVID-19 hastalığı teşhisine yönelik tahminleme işlemi yapar. Şekil 4.4'te gösterildiği gibi "veri seti" dosyasını yüklemek için gerekli kütüphaneleri ve komutları yazar.

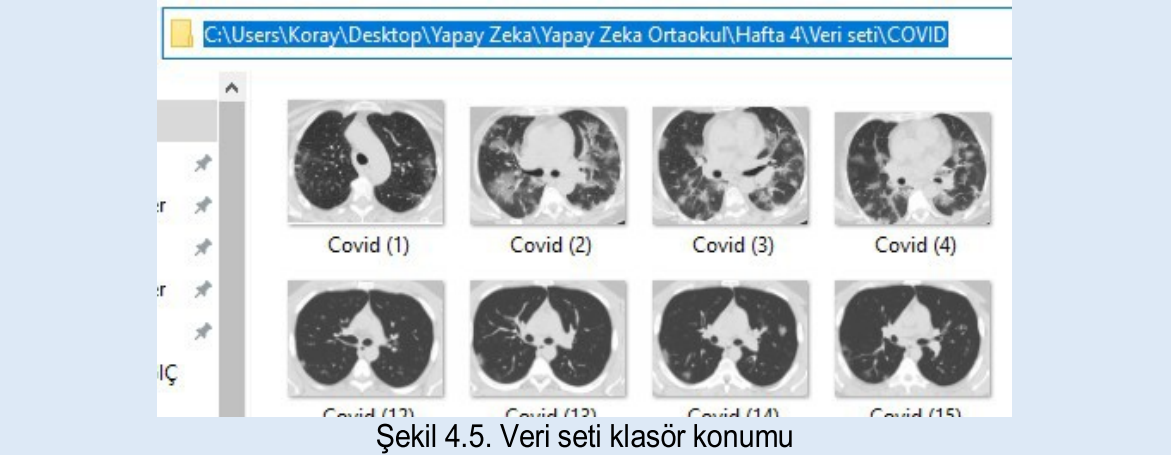


Şekil 4.4. Gerekli kütüphaneleri ve komutların kod gösterimi

Eğitmene Not

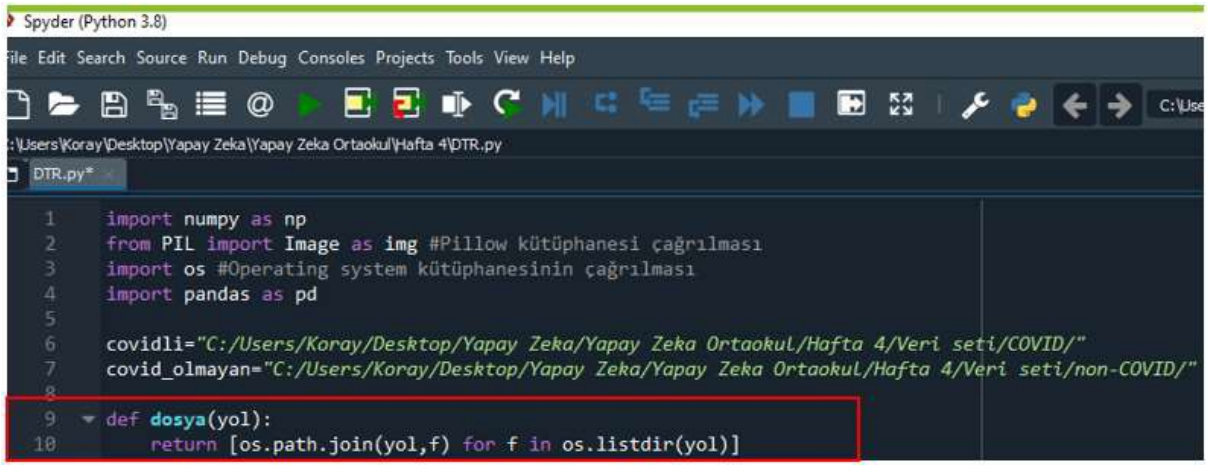
Eğitmen, öğrencilere COVID-19 hastalığı ile ilgili hazırlanacak örnekte ilk olarak dizi işlemlerinde kullanılan "**NUMPY**" kütüphanesinde, bilgisayarlı görü işlemlerinde kullanılan "**PILLOW (PIL)**" kütüphanesinden ve verilerin çekileceği klasörler ile ilgili işlemlerde kolaylık sağlayan "**OS**" kütüphanesinden ve veri setlerinin analizlerinde kullanılan "PANDAS" kütüphanesinden kısaca bahseder.

Eğitmen, Python programlama dilinde 6. ve 7. satırda kod yazarken veri seti klasörü konumunun her bir öğrencinin bilgisayarında farklılık göstereceğini bilir. Bu nedenle eğitmen, öğrencilere Şekil 4.5'te gösterildiği gibi veri setinin konum adresini belirlemeyi öğretir.



Şekil 4.5. Veri seti klasör konumu

Öğrenciler Şekil 4.6'da görüldüğü gibi operating system kütüphanesini kullanarak COVID-19 ve COVID-19 olmayan görüntülerin yer aldığı klasördeki tüm BT görüntülerini döngü kurarak (for döngüsü) bulunduğu klasör konumlarını yol değişkeninden okur.



Şekil 4.6. BT görüntülerinin konumları liste halinde çağırımı kod ekranı

Şekil 4.7'de gösterildiği gibi COVID-19 olmayan verileri dönüştürmek için **"klasor_adi"** ve **"sinif_adi"** parametrelerini içeren **"veri_donusturme"** isminde bir fonksiyon tanımlar.

14. kod satırında COVID-19 ve COVID-19 olmayan görüntüleri bulunduğu klasörlerden tek tek okumak için **"goruntuler"** isminde bir değişken oluşturulur.

16. Kod satırında COVID-19 ve COVID-19 olmayan görüntüleri etiketlemek için **"goruntu_sinif"** isminde boş bir dizi oluşturulur.

17. kod satırında COVID-19 ve COVID-19 olmayan tüm görüntüleri işlemek için bir "**for**" döngüsü oluşturulur.

18. Satırda tüm BT görüntüleri okunarak **".convert('L')"** özelliği kullanılarak gri tonlamalı görüntülere dönüştürülür.

19. Kod satırında **"goruntu_boyutlandirma"** değişkeni kullanılarak tüm BT görüntüleri **".resize"** komutu kullanılarak 28x28 piksel boyutuna küçültülür.

Eğitmene Not

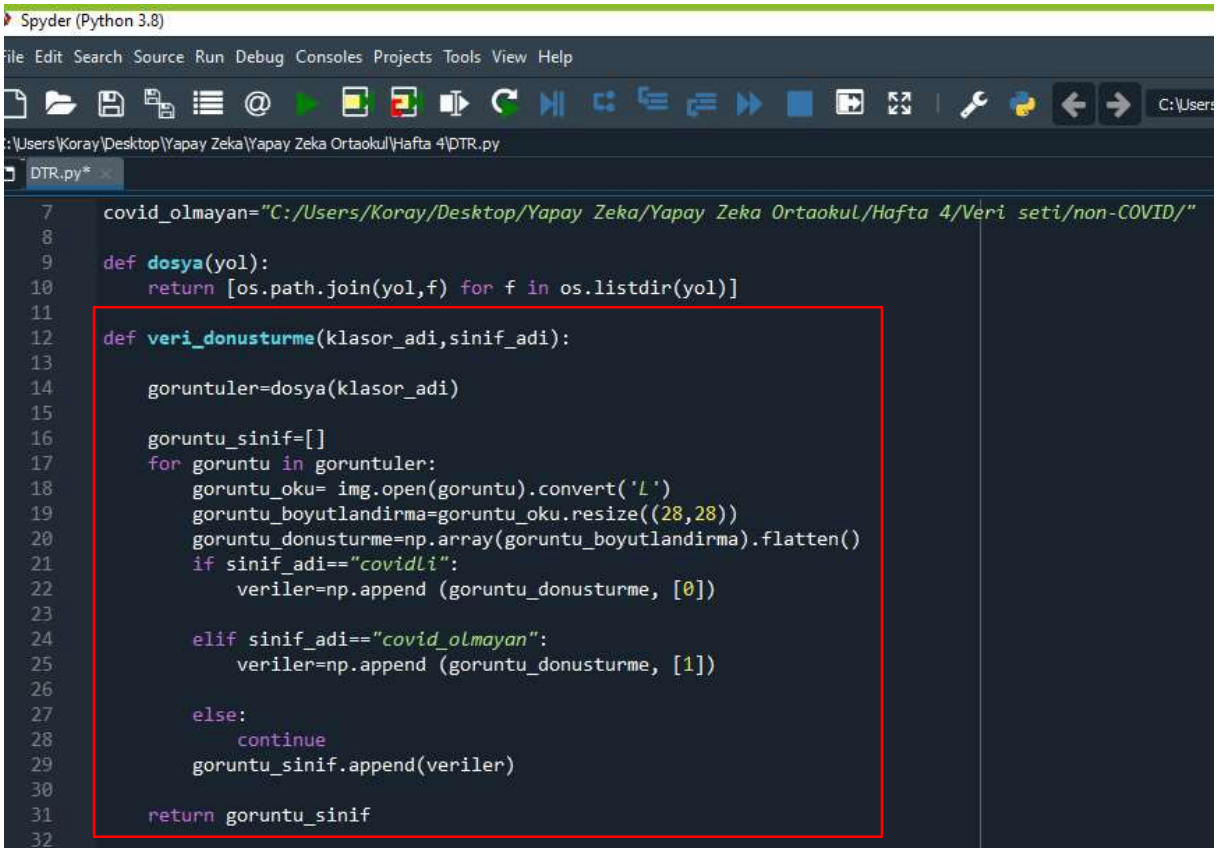
Eğitmen, eğitim sürecini hızlı bir şekilde gerçekleştirmek için BT görüntü boyutunu 28x28 boyutuna düşürdüğünü öğrenciyle paylaşır.

20. Kod satırında iki boyutlu olan BT görüntülerini karar ağaçları ile eğitebilmek için **".flatten"** özelliği kullanılarak görüntüler 784 elemanlı tek boyutlu bir diziye (vektör) dönüştürülür.

Eğitmene Not

Eğitmen, öğrenciye 784 rakamının nasıl elde edildiğini sorar ve tartışır. 784 rakamı 28x28 görüntü boyutuna düşürülen iki boyutlu bir görüntü dizisinin tek boyuta düşürülmesi ile 28x28=784 elde edilir.

21-29 kod satırları arasında, COVID-19 ve COVID-19 olmayan BT görüntüleri etiketlenmiştir. Etiketleme işleminde COVID-19 olanlar **"0"**, COVID-19 olmayan ise **"1"** şeklindedir. COVID-19 ve COVID-19 olmayan bir BT görüntüsü geldiğinde ise etiketleme yapılmamıştır. 31. Kod satırında ise etiketleme yapılan değişken olan **"goruntu_sinif"** değişkeni ile etiketleme işlemi tamamlanmıştır.



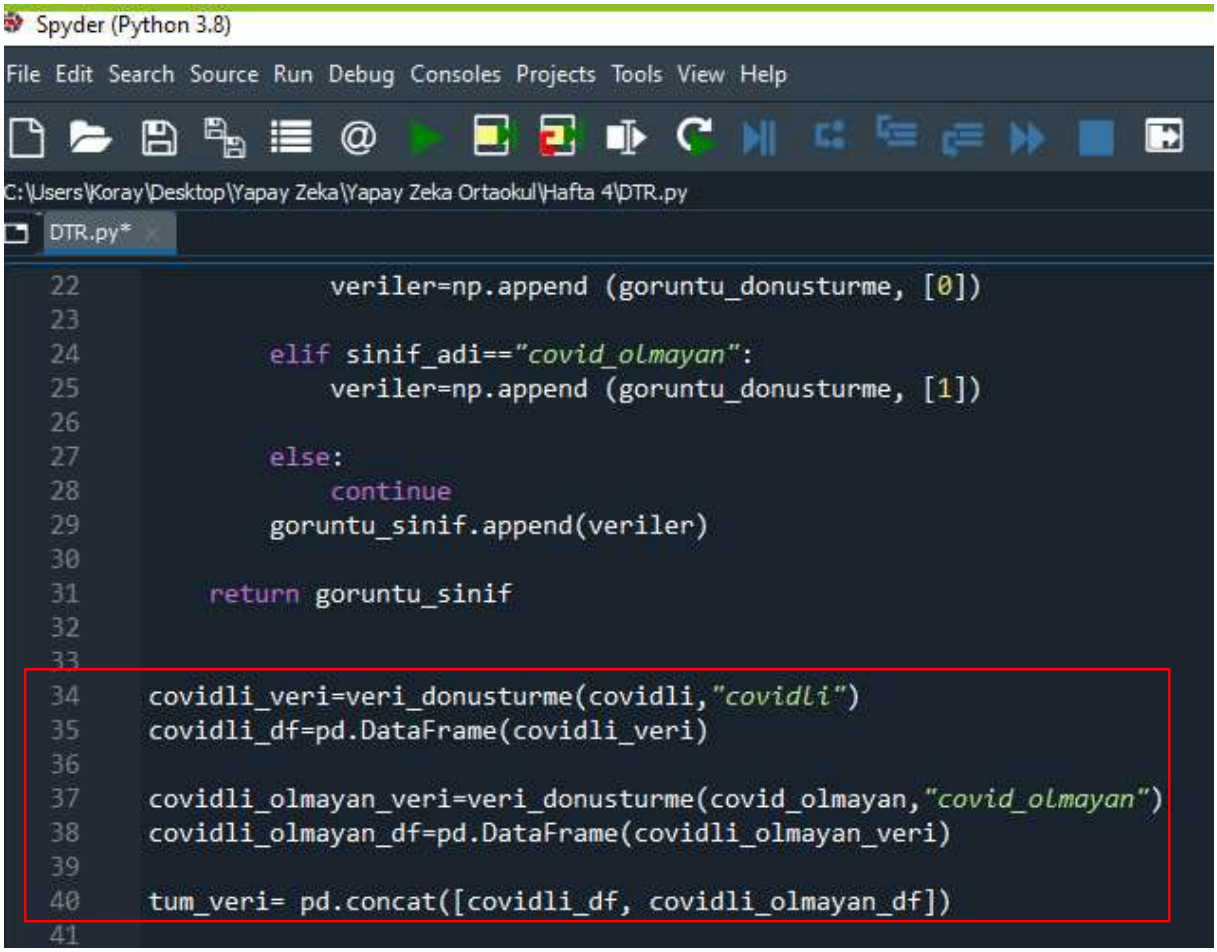
Şekil 4.7. Veri setindeki metin değerlerinin sayısallaştırılması kod ekranı

Öğrenciler Şekil 4.8'de gösterilen 34. kod satırında **"covidli"** veri değişkeni ile oluşturulmuş durumdaki **"veri_donusturme"** fonksiyonuna giderek covidli olarak etiketlenen verileri **"covidli_veri"** değişkenine aktarır.

35. kod satırında ise covidli veriler pandas kütüphanesinin **".DataFrame"** özelliği ile daha sade ve anlaşılabilir hale dönüştürülmüştür.

37. ve 38 kod satırlarında, 34. ve 35. kod satırlarında covidli veriler için yapılan işlemler covidli olmayanlar içinde gerçekleştirilmiştir.

40. Kod satırında ise 34., 35. ve 37., 38. kod satırlarındaki covidli ve covid_olmayan veriler Pandas kütüphanesinin **".concat"** özelliği kullanılarak birleştirilerek **"tüm_veri"** değişkenine aktarılmıştır.



Şekil 4.8. BT görüntülerin etiketlenerek birleştirilmesi

Öğrenciler 43. kod satırında pandas kütüphanesini çağırır.

44. Kod satırında "sklearn" kütüphanesi içerisinde yer alan "**DecisionTreeClassfier**" ile karar ağaçları algoritmalarıyla eğitim gerçekleştirebilmek için **"sklearn.tree"** kütüphanesini çağırır.

45. kod satırında ise, veri setindeki verileri eğitim ve test olarak ayırabilmek için **"sklearn.model_selection"** kütüphanesinde yer alan **"train_test_split"** kütüphanesini çağırır.

46. Kod satırında "sklearn" kütüphanesi içerisinde yer alan "metrics" kütüphanesi kullanılarak model değerlendirilir.

47. Kod satırında ise "**sklearn.model_selection"** kütüphanesinde yer alan **"GridSearchCV"** algoritmasını çağırır.

Eğitmene Not

Eğitmen, öğrenciye **"GridSearchCV"** algoritması hakkında bilgi sahip olup olmadığını sorar ve tartışır?
"GridSearchCV" algoritması modelin hiper parametreleri değiştirilerek model doğruluğunu arttırmak için kullanılan bir algoritmadır.

49. Ve 50. Kod satırlarında ise "**.flatten"** özelliği ile 784 elemanlı tek boyutlu bir vektöre dönüştürülen görüntüler **"Giris"** ve **"Cikis"** değişkenlerine aktarılmıştır.

51. satırda ise, veri setindeki eğitim ve test verilerini %80 eğitim, %20 test olacak şekilde **"test_size=0,2"** komutunu kullanarak rastgele ayırır. Burada "**random_state=1"** ifadesi her eğitim tekrarında alınacak verilerin aynı kalmasını sağlar.

```
43  import pandas as pd
44  from sklearn.tree import DecisionTreeClassifier
45  from sklearn.model_selection import train_test_split
46  from sklearn import metrics
47  from sklearn.model_selection import GridSearchCV
48
49  Giris=np.array(tum_veri)[:,:784]
50  Cikis=np.array(tum_veri)[:,784]
51  Giris_train, Giris_test, Cikis_train, Cikis_test = train_test_split(Giris, Cikis, test_size=0.2, random_state=1)
52
```

Şekil 4.9. Veri setinin eğitim ve test olarak rastgele ayrılması için kod ekranı

Kararlar Veren Makinelerin Çağı!

Düşün, tartış…

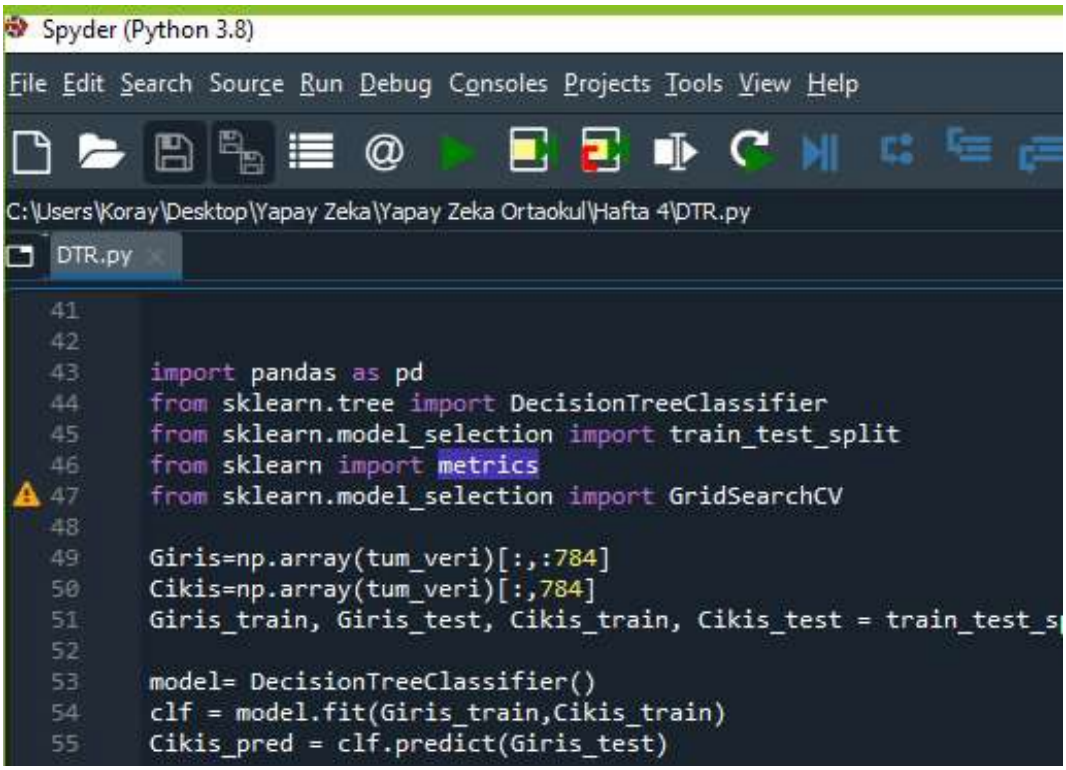
Yapay Zekâ sağlık alanında başarılı teşhisleriyle biliniyor… Sizce Yapay Zekâ'nın kararlarını doğrudan uygulamalı mıyız yoksa son karar verici insanlar mı olmalı? Tartışalım!

4. YÜRÜT

Öğrenciler Şekil 4.10'da gösterildiği gibi 53. kod satırında karar ağaçları algoritmasını **"DecisionTreeClassifier"** sınıflandırıcı ile model değişkenine aktarır.

54. Kod satırında ise BT görüntülerine göre COVID-19 hastalığının tahmini durumu için oluşturmuş olduğu karar ağaçları algoritmasını kullanabilmek için **"model.fit"** komutu yazarak giriş eğitim verilerine göre çıkış eğitim verileri için eğitim gerçekleştirip sonucu **"clf"** değişkenine aktarır.

55. Kod satırında ise karar ağaçları modelinden elde edilen eğitim sonuçlarını **"Giris_test"** verilerine göre **".predict"** özelliği kullanılarak tahmin edip, sonucu **"Cikis_pred"** değişkenine aktarır.



Şekil 4.10. Karar ağaçları algoritmasında eğitim ve tahminleme işlemleri için kod ekranı

5. KARAR VER

5.1. KARAR AĞACI MODELİN BAŞARIM ORANI

Öğrenciler yürüt aşamasında eğitmiş oldukları karar ağaçları modeline ait doğruluk sonucunu ve grafiklerini bu aşamada gerçekleştirir (İlgili kodun tamamı Github platformunda, Hafta 4 altında H4_kararagaci_covid_1.py dosyası olarak paylaşılmış durumdadır. İndirilen veriler, verilen koda göre covid_veri-seti adlı bir klasör altında yer almalıdır). 57. Kod satırında karar ağaçları ile eğitilen BT görüntülerinin tahmin sonuçlarını konsol ekranına aktaran ilgili kodu yazar.

59. kod satırında grafik çizmek için **"matplotlib.pyplot"** kütüphanesini çağırır.

60. Kod satırında ise, **"Cikis_test"** verilerini **"Cikis_pred"** ile karşılaştırıp ROC eğrisini çizebilmek için **".roc_curve"** özelliği kullanılarak yanlış pozitif oranı (ypo), doğru pozitif oranı (dpo) ve threshold (eşik değeri) parametrelerini belirler.

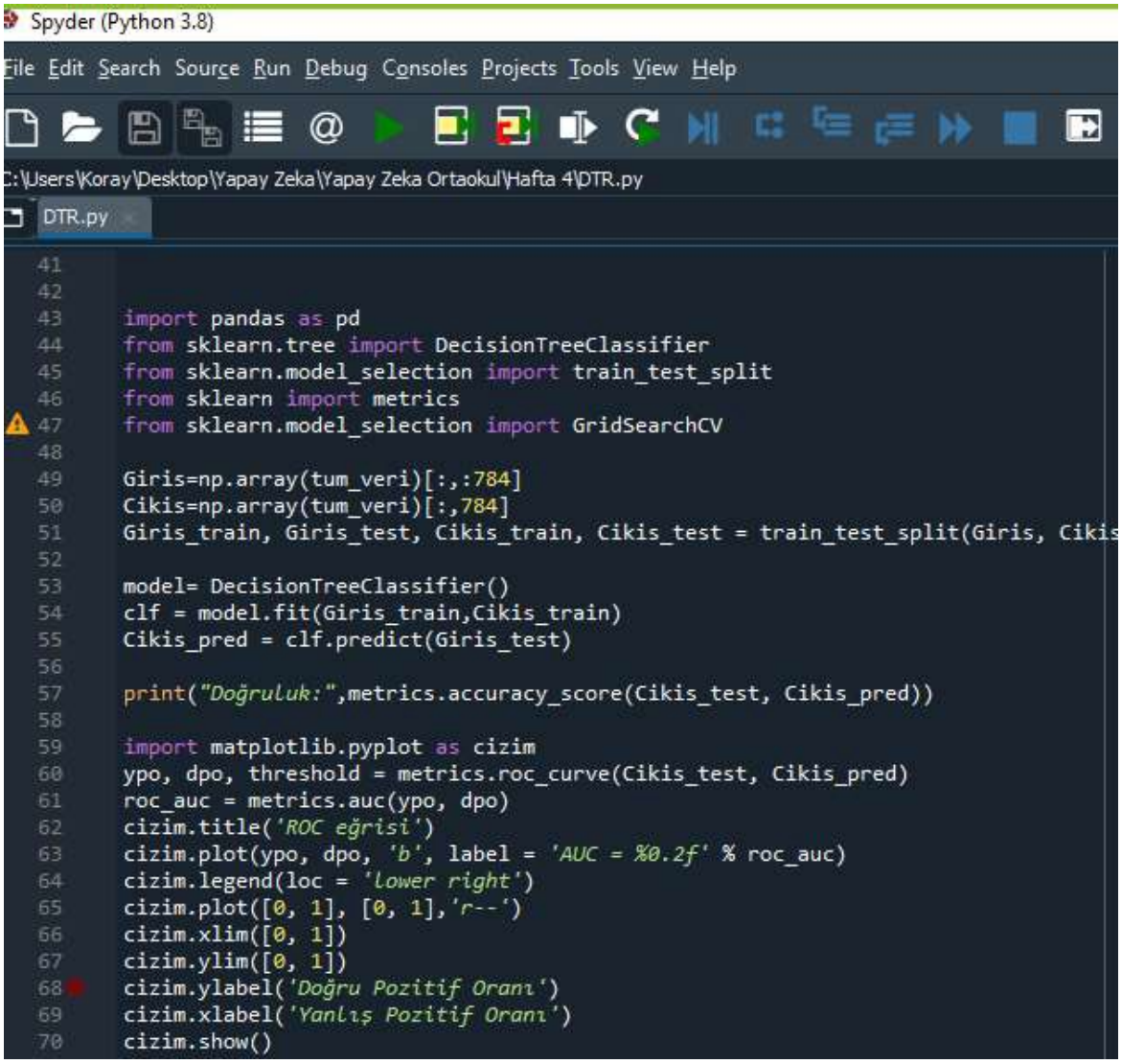
61. Kod satırında ise metrics kütüphanesinin "**.auc"** özelliğini kullanarak yanlış pozitif oranı (ypo) ve doğru pozitif oranı (dpo) değerlerini hesaplayarak "**roc_auc"** değişkenine aktarır.

Eğitmene Not

Eğitmen, öğrencilere yapay zekâda iki veya daha fazla sınıflandırma modelinin performansının ölçümünde doğruluk, kesinlik duyarlılık, f-puanı ve alıcı çalışma karakteristik (Receiver Operating Characteristic-ROC) eğrisini anlatır. Sınıfılandırma modellerinde kullanılan dört farklı durum şu şekildedir:

- Doğru Pozitif Oranı (DPO): Doğru olarak belirlenen pozitif sınıfların tahmin sayısı,
- Doğru Negatif Oranı (DNO): Doğru olarak belirlenen negatif sınıfların tahmin sayısı,
- Yanlış Pozitif Oranı (YPO): Yanlış olarak belirlenen pozitif sınıfların tahmin sayısı,
- Yanlış Negatif Oranı (YNO): Yanlış olarak belirlenen negatif sınıfların tahmin sayısı

Öğrenciler, 62-70. satırları arasında ise ROC eğrisi için başlık, yanlış pozitif oranın (ypo), doğru pozitif oranına (dpo) göre çizimi, etiketlenmesi, hangi renkte çizileceği, hangi sayısal aralıkta çizileceği ve x ve y eksen isimlerine göre ROC eğrisini çizen komutları yazar.

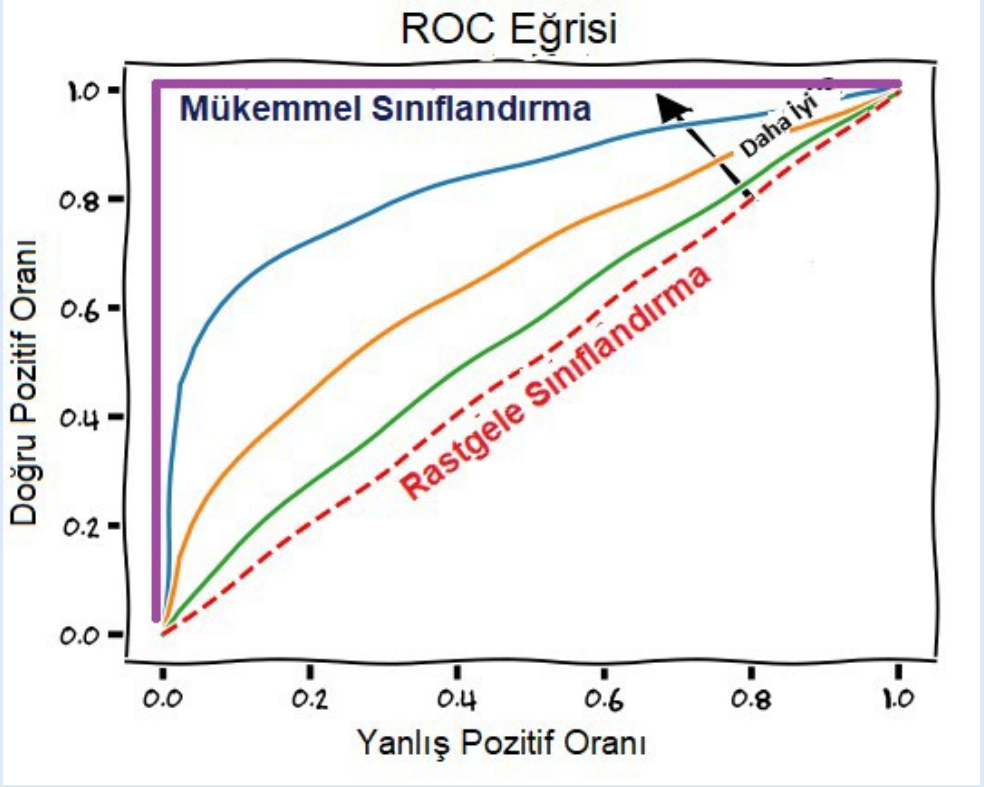


Şekil 4.11. ROC eğrisinin çizimi için kod ekranı

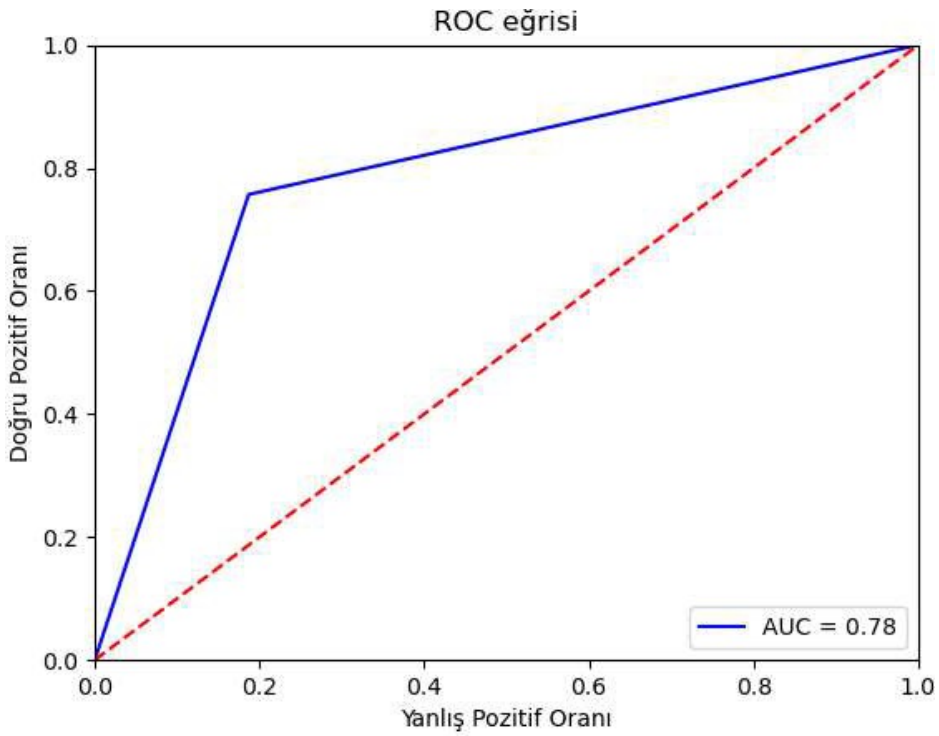
Öğrenciler çeşitli BT görüntülerine göre COVID-19 hastalığı olup olmadığını karar ağaçları algoritması ile tahmin edebilen uygulama örneği için Python kodlarını yazar. Öğrenciler hazırlamış oldukları Python kodlarını çalıştırdıklarında Şekil 4.13'te ve Şekil 4.14'te gösterildiği gibi ROC eğrisini ve modelin başarısını (doğruluk oranı) görüntülerler. Böylece, karar ağaçları algoritması sayesinde, 100 görüntü içerisinde 78 görüntünün doğru başarım oranı (%78) ile tahminlendiğini görürler.

Eğitmene Not

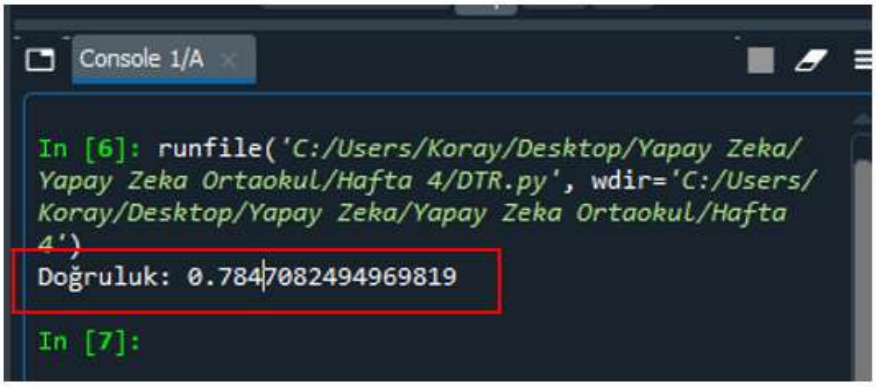
Eğitmen, Şekil 4.12'de gösterildiği gibi öğrenciye çalışma karakteristik eğrisi (Receiver Operating Characteristic-ROC) hakkında açıklama yapar.



Şekil 4.12. ROC Eğrisi gösterimi



Şekil 4.13. Elde edilen ROC eğrisi



Şekil 4.14. Elde edilen doğruluk oranı

Biliyor musunuz?

Karar Ağaçları, Yapay Sinir Ağları ile birlikte Yapay Zekâ'nın Makine Öğrenmesi alanı kapsamındaki en eski tekniklerinden biridir. Tabi ki birçok teknik gibi bu teknik de zamanla birçok çeşitlere ayrılmış ve güncelliğini yitirmemiştir.

Dünya'dan Haberler

Yapay Zekâyla COVID19 (Koronavirüs) Teşhisi

COVID19 pandemisinin başladığı günden bu yana en büyük problemlerden biri de PCR testlerinde bazen yüzde 60'lara varan oranda yalancı negatif sonuçların çıkması oldu. Türkiye'de bu sorun büyük ölçüde, belirtisi olan ama PCR'ı negatif olan hastaya, bilgisayarlı tomografi (BT) ile kesin tanı konarak aşıldı. Ancak radyologların "insan gözüyle" onlarca akciğer BT görüntüsünü okuyup raporlaması, binlerce hastanın söz konusu olduğu bir salgında, sağlık çalışanlarına ekstra yük de getirdi.

Pandeminin başladığı ilk günlerden itibaren yapay zekâ ile hastalığın teşhisi için çalışmalara başladıklarını söyleyen İstanbul İl Sağlık Müdürlüğü Görüntüleme Hizmetleri Koordinatörü Prof. Dr. Muammer Hakkı Karakaş, Sağlık Bilimleri Üniversitesi ve Ankara Üniversitesi Teknokentleri iş birliği ile akciğer tomografisinden Covid teşhisini birkaç saniye içinde "yüzde 99,9 doğrulukla" koyabilen yapay zekâ geliştirildiklerini açıkladı. Projenin sadece teşhis koymakla kalmayıp hekimlere hastalığın, hastadaki gidişatı hakkında öngörü de sunacağını anlatan Prof. Dr. Karakaş, bunun, pandemi yönetimi açısından da önemli veriler sunabileceğini belirterek, "*Yapay zekâ algoritmamız hastanelerin bilgi sistemlerinden hastalarla ilgili birçok datayı da sentezleyerek hastalığın sonlanımı yani gidişatı üzerine bir öngörüde bulunuyor. Yani hangi hastalar hastanede yatacak, hangileri yoğun bakımda yatacak, ya da hangileri ne ölçüde iyileşecek, bunları da artık önceden tespit etmek mümkün. Tomografi çekildikten sonra en nihai kararı yine radyologlar verecek.*" Dedi (Web Kaynağı 4.4).

PYTHON KODLARI (Doğruluk %78):

```
import numpy as np
import PIL.Image as img
import os
import pandas as pd

covidli="covid_veri-seti/COVID/"
covid_olmayan="covid_veri-seti/non-COVID/"

def dosya(yol):
    return [os.path.join(yol,f) for f in os.listdir(yol)]

def veri_donusturme(klasor_adi,sinif_adi):
    goruntuler=dosya(klasor_adi)
    goruntu_sinif=[]
    for goruntu in goruntuler:
        goruntu_oku= img.open(goruntu).convert('L')
        gorunu_boyutlandirma=goruntu_oku.resize((28,28))
        goruntu_donusturme=np.array(gorunu_boyutlandirma).flatten()
        if sinif_adi=="covidli":
            veriler=np.append (goruntu_donusturme, [0])
        elif sinif_adi=="covid_olmayan":
            veriler=np.append (goruntu_donusturme, [1])
        else:
            continue
        goruntu_sinif.append(veriler)
    return goruntu_sinif

covidli_veri=veri_donusturme(covidli,"covidli")
covidli_df=pd.DataFrame(covidli_veri)
```

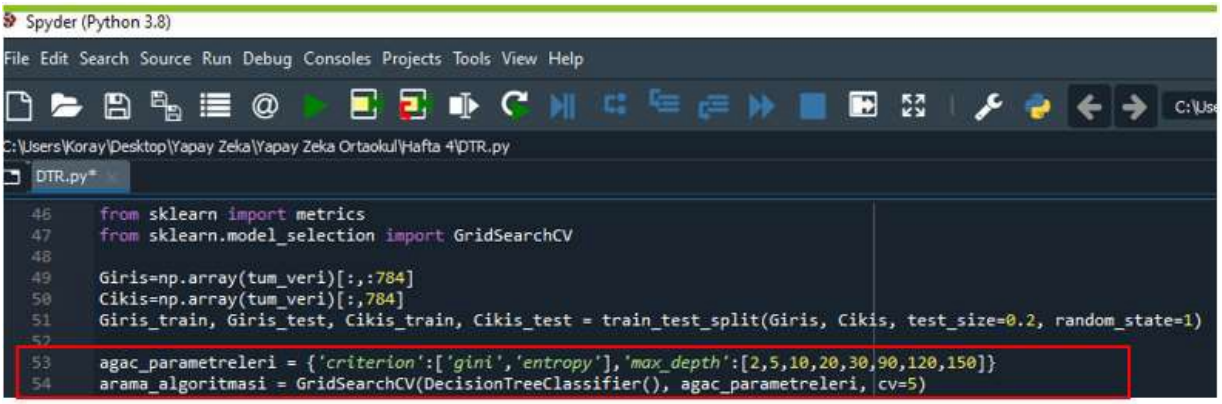
```
covidli_olmayan_veri=veri_donusturme(covid_olmayan,"covid_olmayan")
covidli_olmayan_df=pd.DataFrame(covidli_olmayan_veri)

tum_veri=  pd.concat([covidli_df,  covidli_olmayan_df])

import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.model_selection import GridSearchCV

Giris=np.array(tum_veri)[:,:784]
Cikis=np.array(tum_veri)[:,784]
Giris_train, Giris_test, Cikis_train, Cikis_test = train_test_split(Giris, Cikis,
test_size=0.2, random_state=1)

model= DecisionTreeClassifier()
clf = model.fit(Giris_train,Cikis_train)
Cikis_pred = clf.predict(Giris_test)

print("Doğruluk:",metrics.accuracy_score(Cikis_test,    Cikis_pred))

import matplotlib.pyplot as cizim
ypo, dpo, threshold = metrics.roc_curve(Cikis_test, Cikis_pred)
roc_auc = metrics.auc(ypo, dpo)
cizim.title('ROC  eğrisi')
cizim.plot(ypo, dpo, 'b', label = 'AUC = %0.2f' % roc_auc)
cizim.legend(loc = 'lower right')
cizim.plot([0, 1], [0, 1],'r--')
cizim.xlim([0, 1])
```

```
cizim.ylim([0, 1])
cizim.ylabel('Doğru Pozitif Oranı')
cizim.xlabel('Yanlış Pozitif Oranı')
cizim.show()
```

5.2. MODELİN BAŞARIM ORANININ YÜKSELTİLMESİ

Önceki uygulamada elde edilen başarım oranını yükseltmek için kod üzerinde birtakım güncellemeler gerçekleştirilir (İlgili kodun tamamı Github platformunda, Hafta 4 altında H4_kararagaci_covid_2-h.py dosyası olarak paylaşılmış durumdadır.). Buna göre öğrenciler, 53. kod satırında **"gini"** ve **"entropy"** kriterlerine göre karar ağaçları algoritmasında maksimum derinlik **"max_depth"** özelliği 2 ile 150 arasında sekiz (8) farklı değer alarak sonuçlar **"agac_parametreleri"** sözlüğüne aktarılmıştır. Burada **"gini"** ve **"entropy"** kriterleri, sekiz farklı maksimum derinlik değeri için toplam onaltı kez eğitilmiştir. Maksimum derinlik değerinin 2 ile 150 arasında sınırlandırılmasının nedeni ağaç yapısının karmaşık bir yapıya dönüşmeden eğitimin hızlı biçimde gerçekleştirilmesi ve over fitting'i (aşırı öğrenme) engellemektir.

54. kod satırında ise, GridSearchCV algoritması kullanılarak **"ağaç_parametreleri"** sözlüğünde yer alan veriler 5 parçaya ayrılarak (yani bütün veri %100 olarak kabul edilirse, verileri rastgele %0-20, %20-40, %40-60, %60-80 ve %80-100 oranlarında parçalara bölerek) en iyi sonucu veren parametre değerleri karar ağaçları algoritmasının eğitiminde kullanılır.

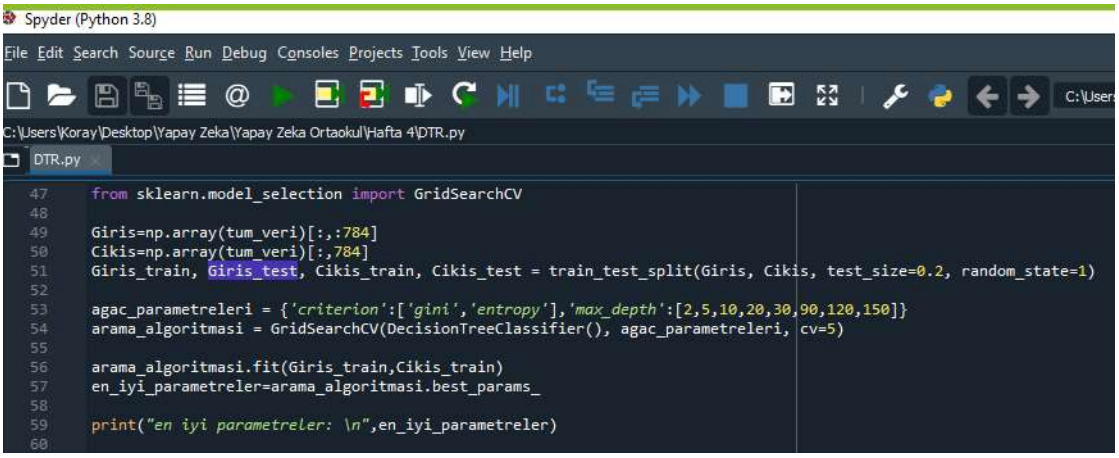


Şekil 4.15. Karar ağacı hiper parametrelerin tanımı ve GridSearchCV algoritmasının uygulanması

Şekil 4.16'da gösterildiği gibi 56. Kod satırında BT görüntülerine göre COVID-19 hastalığının tahmini durumu için oluşturulan karar ağaçları algoritması kullanılarak **"arama_algoritmasi.fit"** komutu yazılarak giriş eğitim verilerine göre çıkış eğitim verilerinin eğitimi gerçekleştirilir.

57. kod satırında ise, GridSearchCV algoritmasının **".best_params_"** özelliği kullanılarak en iyi hiper parametreleri "**en_iyi_parametreler"** değişkenine aktarılır.

59. Kod satırında ise GridSearchCV algoritmasındaki en iyi hiper parametreler konsol ekrana yazdırılır.

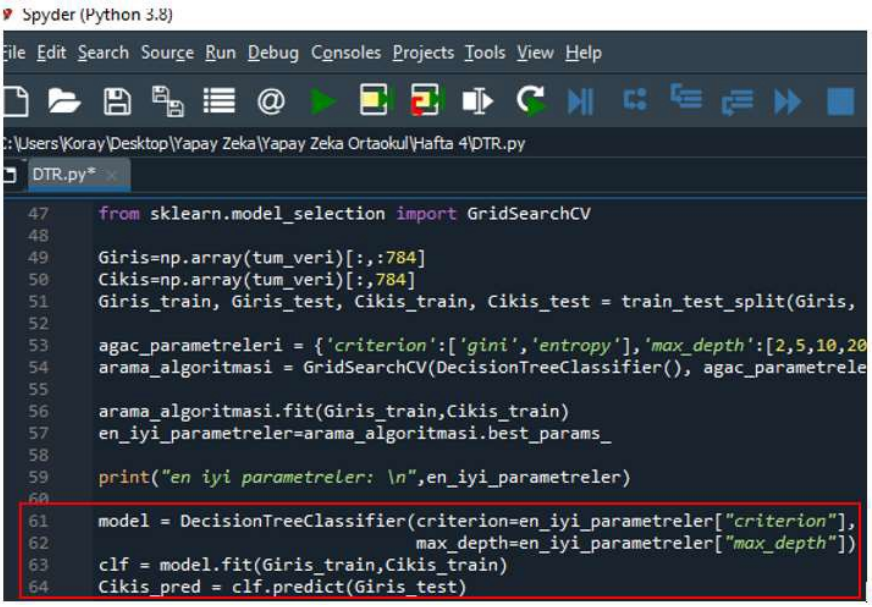


Şekil 4.16. GridSearchCV algoritması ile karar ağaçları modeli için en uygun belirlenen hiper parametrelerin belirlenmesi kod ekranı

Öğrenciler Şekil 4.17'de gösterildiği gibi 61. ve 62. Kod satırında belirlenen kriterler (Gini, Entropy) ve maksimum derinlik (max_depth) parametrelerine göre karar ağacı modeli oluşturur.

63. Kod satırında ise BT görüntülerine göre COVID-19 hastalığının tahmini durumu için oluşturmuş olduğu karar ağaçları algoritmasını kullanabilmek için **"model.fit"** komutu yazarak, giriş eğitim verilerine göre çıkış eğitim verileri için eğitim gerçekleştirip sonucu **"clf"** değişkenine aktarır.

64. Kod satırında ise karar ağaçları modelinden elde edilen eğitim sonuçlarını **"Giris_test"** verilerine göre **".predict"** özelliği kullanılarak tahmin edip, sonucu **"Cikis_pred"** değişkenine aktarır.



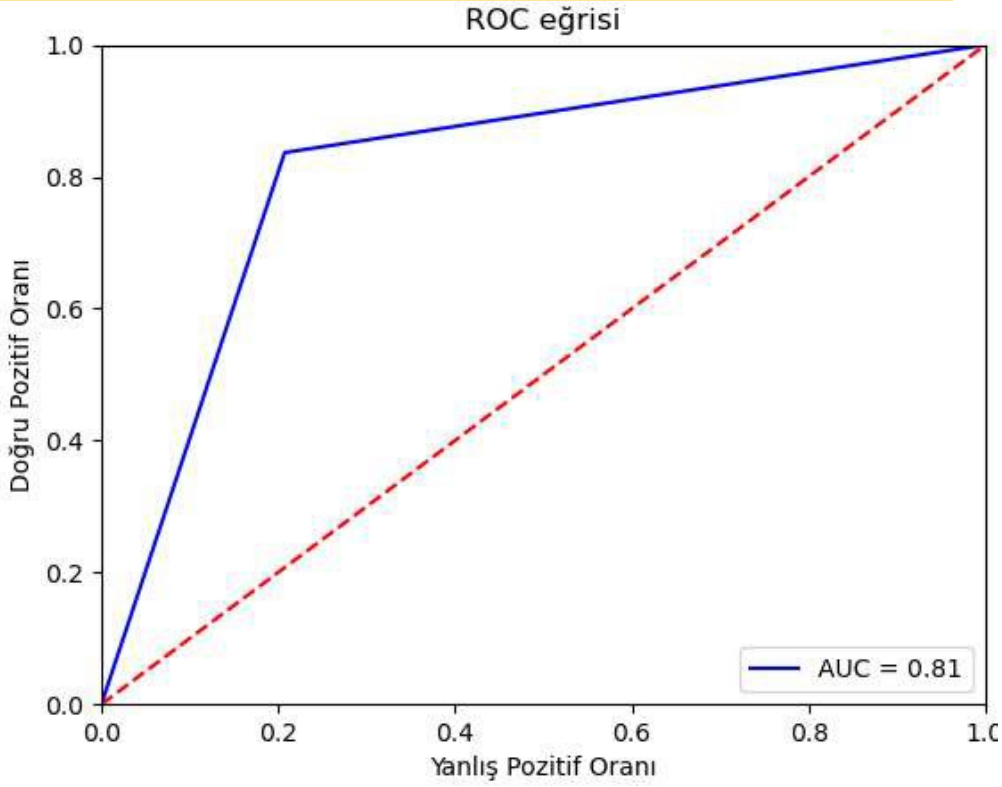
Şekil 4.17. Karar ağaçları algoritmasında eğitim ve tahminleme işlemleri için kod ekranı

PYTHON KODLARI (Doğruluk %81):

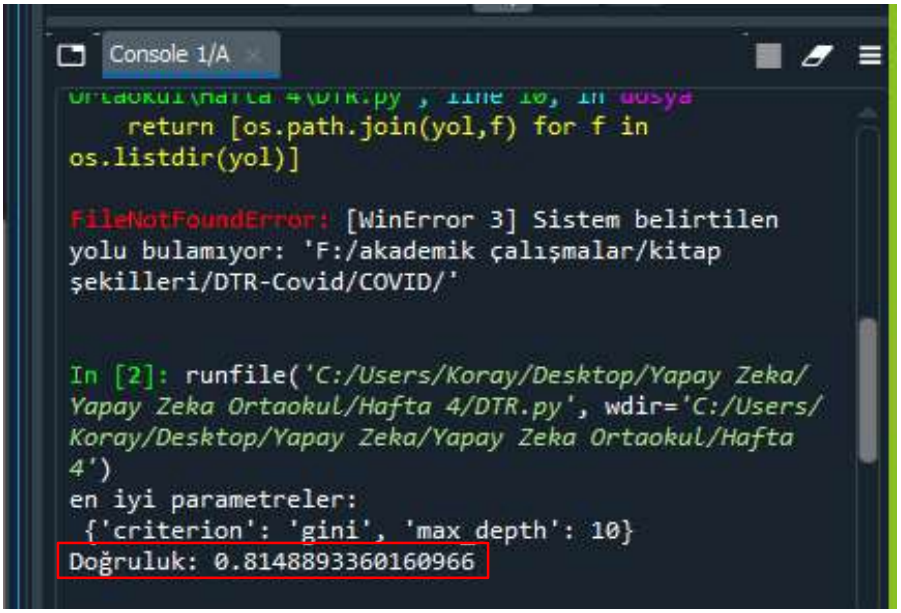
```
import numpy as np
import PIL.Image as img
import os
import pandas as pd

covidli="covid_veri-seti/COVID/"
covid_olmayan="covid_veri-seti/non-COVID/"

def dosya(yol):
    return [os.path.join(yol,f) for f in os.listdir(yol)]

def veri_donusturme(klasor_adi,sinif_adi):
    goruntuler=dosya(klasor_adi)
    goruntu_sinif=[]
    for goruntu in goruntuler:
        goruntu_oku= img.open(goruntu).convert('L')
        gorunu_boyutlandirma=goruntu_oku.resize((28,28))
        goruntu_donusturme=np.array(gorunu_boyutlandirma).flatten()
        if sinif_adi=="covidli":
            veriler=np.append (goruntu_donusturme, [0])
        elif sinif_adi=="covid_olmayan":
            veriler=np.append (goruntu_donusturme, [1])
        else:
            continue
        goruntu_sinif.append(veriler)
    return goruntu_sinif

covidli_veri=veri_donusturme(covidli,"covidli")
covidli_df=pd.DataFrame(covidli_veri)
```

```
covidli_olmayan_veri=veri_donusturme(covid_olmayan,"covid_olmayan")
covidli_olmayan_df=pd.DataFrame(covidli_olmayan_veri)

tum_veri=  pd.concat([covidli_df,  covidli_olmayan_df])

import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.model_selection import GridSearchCV

Giris=np.array(tum_veri)[:,:784]
Cikis=np.array(tum_veri)[:,784]
Giris_train, Giris_test, Cikis_train, Cikis_test = train_test_split(Giris, Cikis,
test_size=0.2, random_state=1)
agac_parametreleri  =
{'criterion':['gini','entropy'],'max_depth':[2,5,10,20,30,90,120,150]}
arama_algoritmasi = GridSearchCV(DecisionTreeClassifier(), agac_parametreleri,
cv=5)

arama_algoritmasi.fit(Giris_train,Cikis_train)
en_iyi_parametreler=arama_algoritmasi.best_params_

print("en iyi parametreler: \n",en_iyi_parametreler)

model = DecisionTreeClassifier(criterion=en_iyi_parametreler["criterion"],
                               max_depth=en_iyi_parametreler["max_depth"])
clf = model.fit(Giris_train,Cikis_train)
Cikis_pred = clf.predict(Giris_test)
```

```
print("Doğruluk:",metrics.accuracy_score(Cikis_test,    Cikis_pred))


import matplotlib.pyplot as cizim
ypo, dpo, threshold = metrics.roc_curve(Cikis_test, Cikis_pred)
roc_auc = metrics.auc(ypo, dpo)
cizim.title('ROC  eğrisi')
cizim.plot(ypo, dpo, 'b', label = 'AUC = %0.2f' % roc_auc)
cizim.legend(loc = 'lower right')
cizim.plot([0, 1], [0, 1],'r--')
cizim.xlim([0, 1])
cizim.ylim([0, 1])
cizim.ylabel('Doğru Pozitif Oranı')
cizim.xlabel('Yanlış Pozitif Oranı')
cizim.show()
```



Şekil 4.18. Elde edilen ROC eğrisi



Şekil 4.19: Karar ağaçları modelinin en iyi kriterler ve maksimum derinlik hiper parametrelerine göre doğruluk oranı

Eğitmene Not

Karar ağaçları algoritması ile eğitim esnasında modellerin doğrulukları birbirinden farklı değerler çıkabilir. Birbirinden farklı değerler çıkması genellikle veri seti ile ilgili bir durumdur. Kod satırında yer alan "random_state=109" kod satırında 109 rakamını değiştirerek (örneğin "0", "1", vs.) modelin doğruluğunun değişliğini gözlemleyiniz ve tartışınız. Eğitmen, en yüksek doğruluğu elde edinceye kadar modeli eğitebilir.

6. İLAVE ETKİNLİK

Şekil 4.20'de gösterildiği gibi orman yangınını işaret eden; duman veya yangın görüntülerine göre orman yangınının olup olmadığını karar ağaçları algoritması kullanarak tahminleyen modeli kurunuz. (Bu modelin ormanları tarayan bir drone içerisine entegre edileceğini düşünün…). İlgili uygulamanın kodu Github platformu Hafta4 klasörü altında H4_kararagaci_orman-yangini.py dosyası ile paylaşılmış durumdadır.)

Şekil 4.20. Orman yangını görüntüsü

VERİ SETİ İÇİN İNDİRME LİNKİ

https://github.com/aiformankind/wildfire-dataset/archive/refs/heads/master.zip

Eğitmene Not

Eğitmen, ilgili indirme linkinden veri setini indirir. Çizelge 4.2'de gösterildiği gibi negative (yangın olmayan görüntüler) ve positive (yangın olan görüntüler) isimli dosyaları ilave etkinlikte kullanılmak üzere öğrencilere verir (Anlatılan kodlara uygun olması için verileri orman-yangini adlı bir klasör altında toplanmasını istenebilir.). Eğitmen, öğrencilere veri setinde yer alan giriş parametresi görüntülere göre çıkış parametresi olan yangın yok ise negatif (0), yangın var ise pozitif (1) sınıfı temsil ettiğini anlatır.

Çizelge 4.2. Veri setinin giriş çıkış parametreleri

Veri Sayısı	GİRİŞ PARAMETRESİ	ÇIKIŞ PARAMETRESİ
	Görüntü	Yangın olup olmadığı
1		0 (Negatif)
2		0 (Negatif)
...	...	...
97		0 (Negatif)

98		1 (Pozitif)
...	...	...
302		1 (Pozitif)

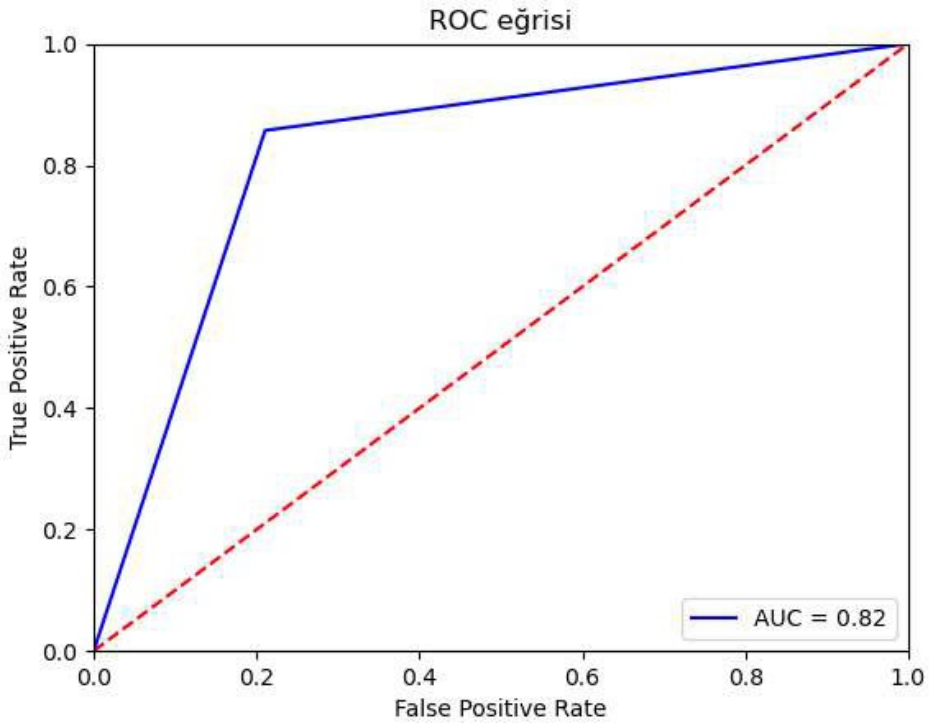
PYTHON KODLARI:

```
import numpy as np
import PIL.Image as img
import os
import pandas as pd

duman_yangin_yok="orman-yangini/negative/"
duman_yangin_var="orman-yangini/positive/"

def dosya(yol):
    return [os.path.join(yol,f) for f in os.listdir(yol)]

def veri_donusturme(klasor_adi,sinif_adi):

    goruntuler=dosya(klasor_adi)

    goruntu_sinif=[]
```

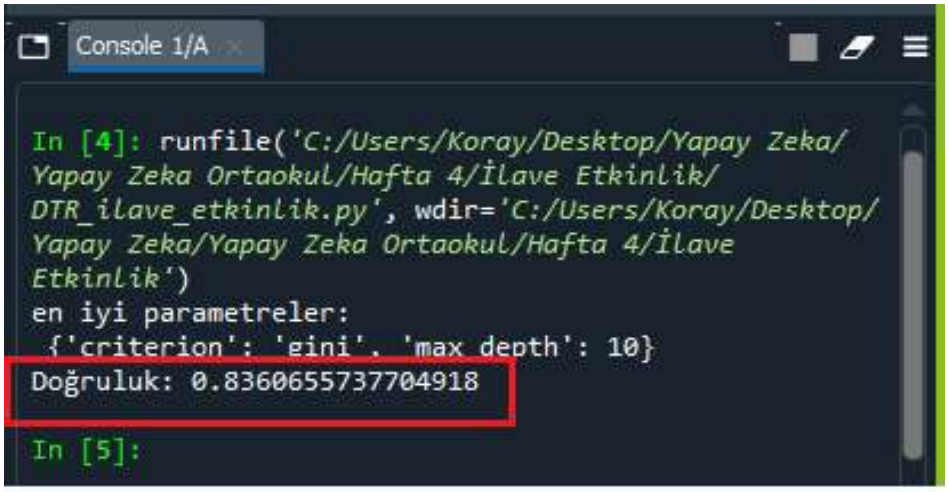
```
    for goruntu in goruntuler:
        goruntu_oku= img.open(goruntu).convert('L')
        gorunu_boyutlandirma=goruntu_oku.resize((28,28))
        goruntu_donusturme=np.array(gorunu_boyutlandirma).flatten()
        if sinif_adi=="yok":
            veriler=np.append  (goruntu_donusturme,  [0])

        elif sinif_adi=="var":
            veriler=np.append  (goruntu_donusturme,  [1])

        else:
            continue
        goruntu_sinif.append(veriler)

    return goruntu_sinif



duman_yangin_yok_veri=veri_donusturme(duman_yangin_yok,"yok")
duman_yangin_yok_df=pd.DataFrame(duman_ates_yangin_veri)

duman_yangin_var_veri=veri_donusturme(duman_yangin_var,"var")
duman_yangin_var_df=pd.DataFrame(duman_yangin_var_veri)

tum_veri=  pd.concat([duman_yangin_yok_df,  duman_yangin_var_df])



import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.model_selection import GridSearchCV
```

```
Giris=np.array(tum_veri)[:,:784]
Cikis=np.array(tum_veri)[:,784]
Giris_train, Giris_test, Cikis_train, Cikis_test = train_test_split(Giris, Cikis,
test_size=0.2, random_state=1)

agac_parametreleri  =
{'criterion':['gini','entropy'],'max_depth':[2,5,10,20,30,90,120,150]}

arama_algoritmasi = GridSearchCV(DecisionTreeClassifier(), agac_parametreleri,
cv=5)
arama_algoritmasi.fit(Giris_train,Cikis_train)
en_iyi_parametreler=arama_algoritmasi.best_params_

print("en iyi parametreler: \n",en_iyi_parametreler)

model = DecisionTreeClassifier(criterion=en_iyi_parametreler["criterion"],
                               max_depth=en_iyi_parametreler["max_depth"])

clf = model.fit(Giris_train,Cikis_train)

Cikis_pred = clf.predict(Giris_test)
print("Doğruluk:",metrics.accuracy_score(Cikis_test, Cikis_pred))

import matplotlib.pyplot as plt
fpr, tpr, threshold = metrics.roc_curve(Cikis_test, Cikis_pred)
roc_auc = metrics.auc(fpr, tpr)
plt.title('ROC eğrisi')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
```

```
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



Şekil 4.21. Elde edilen ROC eğrisi



Şekil 4.22. Ulaşılan doğruluk değeri

Eğitmene Not

Eğitmen, bu haftaki eğitime yönelik öğrencilerin ilgisini arttırmak ve yeni öğrenilen bilgileri önceden öğrenilen bilgiler ile karşılaştırmalarını sağlamak için aşağıda maddeler halinde verilen soruları sınıfta tartışır.

- İlgili haftada ele alınan yapay zekâ uygulamalarına uygun örnek veri setleri neler olabilir?
- İlgili haftada ele alınan veri setlerinin yapay zekâ teknik ve kütüphanelerinde ne derecede başarım etkisi olmuştur?
- İlgili haftada öğretilen yapay zekâ tekniği ile uyumlu ve uyumsuz olabilecek açık erişimli veri seti çeşitleri sunan (kaagle, github vs.) internet sitelerinden elde edilecek farklı veri çeşitleri ile yapay zekâ tekniği uyumluluğunu nasıldır?

.

Eğitmene Not

Eğitmen, öğrencilerin 3. ve 4. Hafta içerisinde elde ettikleri bilgi-becerileri değerlendirmek, aktif katılımlarını sağlamak ve onları gelecek haftalara motive etmek için 'Kahoot' uygulamasını ya da benzeri bir Web 2.0 uygulamasını kullanarak 'bilgi yarışması' düzenleyebilir.

3. ve 4. Hafta bağlamında sorulabilecek sorular; Python ile Makine Öğrenmesi temel çözüm süreçleri, Bayes Öğrenmesi temelleri / uygulamaları ve Karar Ağaçları temelleri / uygulamaları yönünde olabilir.

Kaynakça

SPSS. (1999). AnwerTree Algorithm Summary, SPSS White Paper, USA.

Web Kaynağı 4.1: https://erdincuzun.com/makine_ogrenmesi/decision-tree-karar-agaci-id3-algoritmasi-classification-siniflama/

Web Kaynağı 4.2: https://www.datasciencearth.com/underfitting-ve-overfitting/

Web Kaynağı 4.3: https://www.webtekno.com/avustralya-mahkeme-yapay-zekâ-mucit-olabilecegini-onayladi-h113004.html

Web Kaynağı 4.4: https://www.cnnturk.com/video/turkiye/yapay-zekâyla-koronavirus-teshisi

5. Hafta: Yapay Sinir Hücresi ve Yapay Sinir Ağları

Ön Bilgi:

- Yapay sinir ağları temelleri, çok katmanlı algılayıcı modeli, alternatif yapay sinir ağları modelleri, yapay sinir ağları ile akıllı tarım uygulaması
- Python kodlamada döngüler, fonksiyonlar ve kütüphane yapılarının kullanımı (Bu konuda Python anlatım videoları: 10-11. Döngüler, 13. Fonksiyonlar, 14-16. Python kütüphaneleri serisi yardımcı araç olarak kullanılabilecektir).

Haftanın Kazanımları:

- Öğrenciler yapay sinir ağları temel yapısını kavrar.
- Öğrenciler yapay sinir ağları ile insan beyninin öğrenme yapısını modelleyerek uygular.
- Öğrenciler yapay sinir ağları ile matematiksel modelleme denklemlerini yorumlamayı kavrar.
- Öğrenciler yapay sinir ağları ile pamuk tarlasına ait sulama sistemini modelleme ve çözüm üretme yetenekleri kazanır.
- Öğrenciler yapay sinir ağları üzerindeki bağımlı değişken, bağımsız değişken, ağırlık değeri ve bias terimlerini kavrar.
- Öğrenciler Python programlama dilini kullanarak pamuk tarlasına ait akıllı sulama için program kodunu yazabilir.
- Öğrenciler Python programlama dilinde yapay zekâ modelleri için hazırlanan keras kütüphanesinin fonksiyonlarını kullanma yeteneği kazanır.
- Öğrenciler pamuk tarlasına ait verileri giriş ve çıkış parametresine göre belirler.
- Öğrenciler yapay sinir ağlarında veriye uygun model tasarlar, analiz eder ve modeli eğitir.
- Öğrenciler test verileri üzerinde modeli değerlendirerek yorumlar.
- YSA modeli ile pamuk tarlası sulama sistemi için konsol ekranında %95 olarak yüksek bir başarım oranı ile model oluşturma yeteneği kazanır.
- Yapay Sinir Ağları ile farklı verilerle yanlış öğrenmenin gerçekleşip gerçekleşmeyeceğini kavrar.
- Türkiye'de 2021-2025 Ulusal Yapay Zekâ Stratejisi'nin yürürlüğe girdiğini öğrenir.

Haftanın Amacı:

Bu haftanın amacı, "Yapay Sinir Ağları" kavramının tüm öğrenciler tarafından doğru bir şekilde anlaşılmasını sağlamaktır. Haftanın bir diğer amacı, yapay sinir ağlarının kullanıldığı farklı örnekler ile öğrencilerin dikkatini çekmek ve etkinliklere yönelik ilgilerini artırmaktır. Ayrıca, öğrencilere Python programlama dili kullanarak yapay sinir ağları ile örnek modelleme ve çözüm üretme yetenekleri kazandırılacaktır.

Kullanılacak Malzemeler:

Bilgisayar, kâğıt, kalem, Python kod editörü, örnekler için çıktı görselleri

Haftanın İşlenişi:

Algıla: Yapay sinir ağları kavramlarını öğrenir.

Tasarla: Yapay sinir ağları ile pamuk tarlasına yönelik çözümlenecek tarım uygulamasının giriş ve çıkış parametrelerini belirler. Bu hafta için Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşim Tasarla aşaması ile başlatılabilmektedir.

Harekete Geç: Python ile veri setindeki görüntü değerlerini sayısallaştırıp eğitim ve test verilerini ayırıp model kurar.

Yürüt: Eğitmen öğrenciler ile etkileşimli olarak yapay sinir ağları modelini eğitir ve kişinin akıllı sulama sistemi için pompa motorunun çalışıp çalışmaması durumunu belirler.

Karar Ver: Haftanın içeriğinin Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşimli bir biçimde kullanılır.

1. ALGILA

1.1. Yapay Sinir Ağları Temelleri

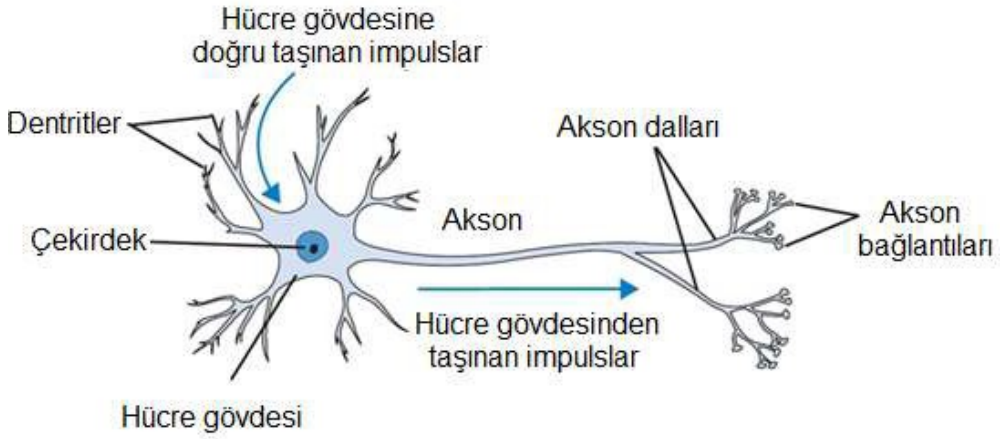
Eğitmene Not

Eğitmen, öğrencilere "**sibernetik**", "**yapay sinir ağları**", "**modelleme**" "**aktivasyon fonksiyonu**" "**yapay sinir ağları katmanları**" ve "**nöron**" kavramları hakkında bilgi sahibi olup olmadıklarını sorar ve tartışır. Böylece öğrencilerin konuya dikkatini çeker.

Canlıların davranışlarını inceleyerek matematiksel olarak modelleyip, benzer yapay modellerin üretilmesine "**sibernetik"** denir. Eğitilebilir, adaptif, kendi kendine organize olup öğrenebilen ve değerlendirme yapabilen yapay sinir ağları ile insan beyninin öğrenme yapısının modellenmesi amaçlanmıştır. Bu bakımdan Yapay Sinir Ağları tekniği, içerisindeki esnek matematiksel yapı sayesinde, Yapay Zekâ'nın en güçlü tekniği olma unvanını elinde tutmaktadır. Yapay sinir ağları vasıtasıyla tıpkı insanoğlunda olduğu gibi makinelerin eğitilmesi, öğrenmesi ve karar vermesi amaçlanmaktadır. Şekil 5.1'de insan sinir hücresinin (nöron) örnek görseli verilmiştir.

Eğitmene Not

Eğitmen, öğrencilere Şekil 5.1'de verilen insan sinir hücresinin yapısını, çalışma prensibini ve özelliklerini detaylı biçimde anlatarak açıklar.

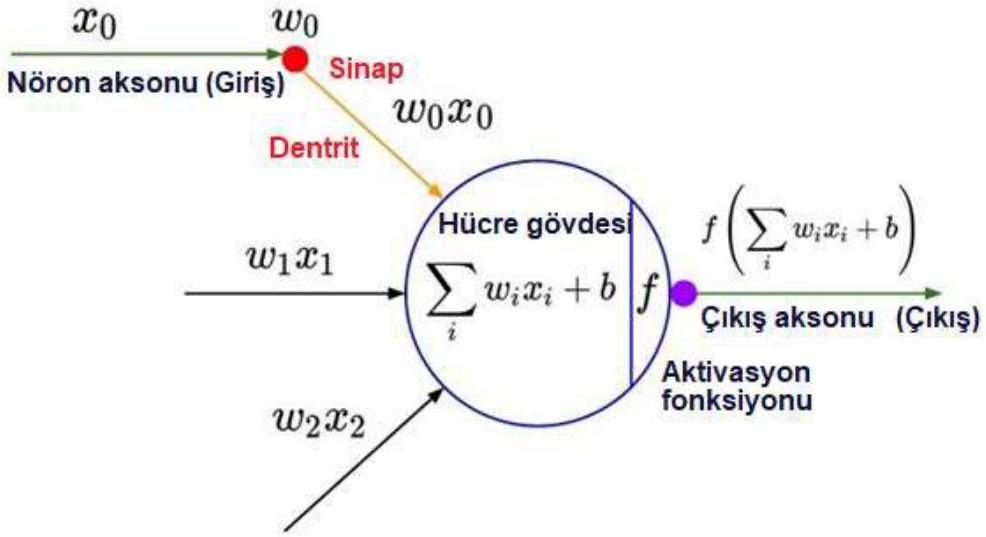


Şekil 5.1. İnsan sinir hücresinin yapısı

Şekil 5.2'de gösterildiği gibi Yapay sinir ağları insan sinir hücrelerine benzer bir yapıda giriş değerlerine göre (x0, x1, x2 vs.) ağırlıkları belirlenip (w0, w1, w2,...vs) çarpılarak elde edilen sonuçların toplanması ilkesine dayanmaktadır. Yine yapay sinir ağlarında sabit bir bias değeri de kullanılabilmekte; bu değer ile aktivasyon fonksiyonlarının sınıflandırma ya da regresyon yönündeki hesaplamaları yönlendirilebilmektedir.

Eğitmene Not

Eğitmen, Yapay Sinir Ağları tekniğindeki temel mantığı, üzerinde çok çalıştığımız bir konuyu pekiştirme yaptıkça hatırlamamızı sağlayan, beyin hücreleri arasındaki artan elektriksel yüke benzeterek somutlaştırmaya çalışır. Bu noktada üzerinde çok düşünülmeyen konuların, hücreler arası elektriksel yüklerin azalması nedeniyle unutulacağını da ifade eder. Yine beyin hücreleri arasındaki elektriksel bağların, Yapay Sinir Ağları içerisindeki hücrede ağırlık değerleriyle karşılandığını açıklayarak, doğal ve yapay sinir hücresi arası esin kaynaklarını irdeler.

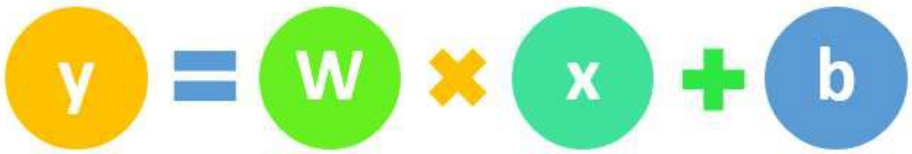


Şekil 5.2. Temel yapay sinir hücresinin matematiksel modeli

Temel bir yapay sinir ağı modelinin matematiksel denklemi Şekil 5.3'te verilmiştir. Burada;

- y: x'e bağlı bağımlı değişken olup, giriş parametrelerine göre modelden elde edilen doğruluk sonucunu verir.
- x: Bağımsız giriş parametresidir.
- w: Her bir giriş parametresinin ağırlık değeridir.
- b: Sabit bir bias değeridir.

Yapay sinir ağı modellerinde w ve b parametreleri değiştirilerek en iyi doğruluk sonucu elde edilinceye kadar model eğitilir.



Şekil 5.3. Yapay sinir ağlarının matematiksel denklemi

Eğitmene Not

Eğitmenin, şu linkteki yapay sinir ağları matematiksel gösterimi ile ilgili bilgileri dersten önce incelemesi önerilir.

1.2. Tek ve Çok Katmanlı Yapay Sinir Ağı Modelleri

YSA tek ve çok katmanlı olmak üzere iki farklı model olarak tasarlanabilir. İlk olarak yapay sinir ağı modelleri tek katmanlı yapıya sahip olarak tasarlanmıştır. 1960 yılında Widrow ve Hoff çok gizli katmanlı yapıya geçen ilk çalışmayı yapmışlardır. Şekil 5.4'te tek ve çok katmanlı yapay sinir ağları modellerinin görseli verilmiştir. Tek gizli katmanlı YSA modelinde giriş katmanlarından sonra oluşturulan gizli katmandaki nöron sayısına göre modeller eğitilerek çıkış katmanına gönderilir. Tek gizli katmanlı YSA modelinde 4+2=6 nöron (çıkış ve gizli katmandaki nöron sayısı) bulunmaktadır. Öğrenilecek olan parametre sayısı;

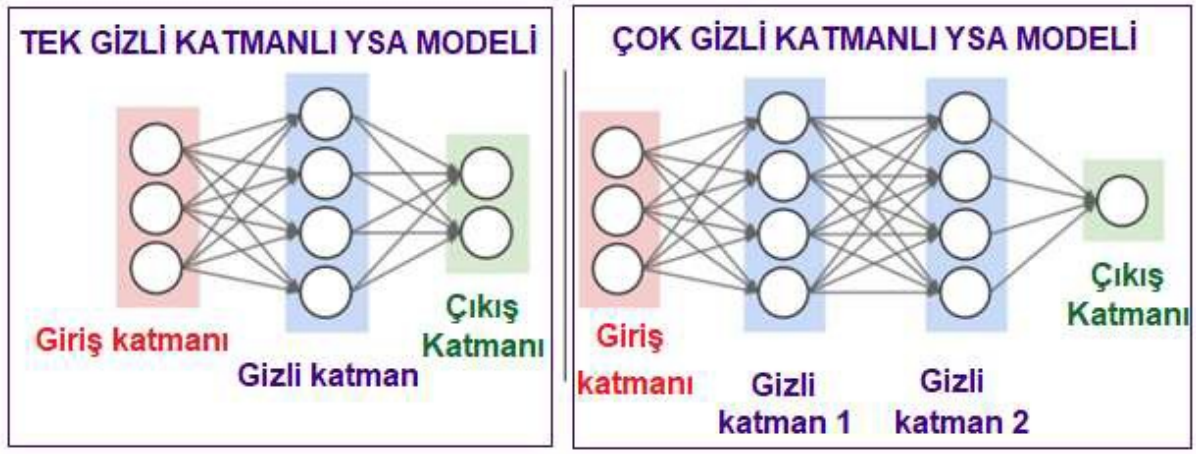
Giriş katmandaki nöron sayısı (3) X gizli katmandaki nöron sayısı (4) + gizli katmandaki nöron sayısı (4) x çıkış katmanındaki nöron sayısı (2) = 20 ağırlık değeri elde edilir.

Elde edilen ağırlık değeri gizli katmandaki nöron sayısı (4) + çıkış katmandaki nöron sayısının (2) toplamı ile elde edilen6 (altı) değeri bias değeri olup eklenerek toplam 26 adet öğrenilmesi gereken parametre hesaplanır.

Çok gizli katmanlı YSA modelinde gizli katmandaki nöron sayısı-1 (4) + gizli katmandaki nöron sayısı-2 (4) + çıkış katmanındaki nöron sayısı (1) olmak üzere 9 nörondan (çıkış ve gizli katmanlardaki nöron sayısı) oluşmaktadır.

Giriş katmandaki nöron sayısı (3) X gizli katman-1 nöron sayısı (4) + gizli katman-1 nöron sayısı (4) x gizli katman-2 nöron sayısı (4) + gizli katman-2 nöron sayısı (4) x çıkış katmanındaki nöron sayısı (1) = 32 ağırlık değeri elde edilir.

Elde edilen ağırlık değeri gizli katman-1 nöron sayısı (4) + gizli katman-2 nöron sayısı (4) + çıkış katmandaki nöron sayısının (1) toplamı ile elde edilen değer 9 (dokuz) değeri bias değeri olup eklenerek toplam 41 adet öğrenilmesi gereken parametre hesaplanır.



Şekil 5.4. Tek ve çok gizli katmanlı yapay sinir ağı model yapıları

Biliyor musunuz?

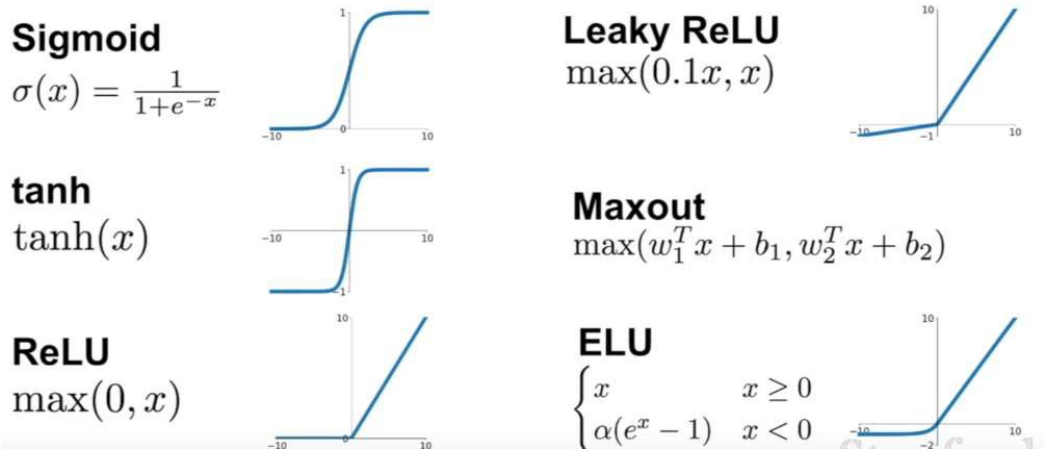
Yapay Sinir Ağları tekniğinin oluşmasına sebep olan önemli gelişmelerden biri de ilk aşamada katmansız kullanılan nöron yapılarının XOR adı verilen bir mantık kapısı problemine (giriş değerleri farklıysa sonuç 1: Doğru, aksi halde 0: Yanlış'tır) iyi sonuç üretememesidir. Yapılan çalışmalar katmanlar altında tasarlanan ağ yapısı halindeki nöronların bu problemi çözebildiğini ('birlikten kuvvet doğar.') göstermiş ve dolayısıyla Yapay Sinir Ağları çağı başlamıştır…

1.3. Yapay Sinir Ağlarında Nöronların Kullanımı

Yapay sinir ağında katmanlar içinde kullanılan nöronların birbirleri ile ilişkisi yoktur. En önemli görevleri sistemde olan bilgiyi bir sonraki katmana ya da çıkış katmanına aktarmaktır. Ard arda gelen iki katmandaki nöronlar farklı aktivasyon değerlerine göre YSA modelin öğrenme seviyesini belirleyip aktarım işlemini gerçekleştirir. Yapay sinir ağlarında nöron sayısı, giriş parametresi ve katman sayısı modelin önemli parametrelerindendir. Pek çok hiper parametre kullanılarak çıkış performansı arttırılabilir. W ağırlık vektörü ise düğüm/nöron sayısı (hücre), bias (b) değerleri de gelecek katmandaki düğüm sayısı kadar olmalıdır.

1.4. Yapay Sinir Ağlarında Aktivasyon Fonksiyonları

Yapay sinir ağı modelinde regresyon veya sınıflandırma probleminin çözümü için kurulan modelde farklı aktivasyon fonksiyonları kullanılabilir (Şekil 5.5).



Şekil 5.5. Yapay sinir ağlarında sıklıkla kullanılan aktivasyon fonksiyonları

1.4.1. Sigmoid Fonksiyonu

YSA modellerinde sıklıkla kullanılan aktivasyon fonksiyonlarından birisi de Sigmoid fonksiyonudur. Sigmoid fonksiyonu [0,1] arasında değerler alabilen ve sınıflandırma problemlerinde kullanılan bir fonksiyondur. Bu fonksiyon, modelden elde edilen sonucun hangi sınıfa ait olduğunu öğrenen, olasılıksal bir değer üretir.

1.4.2. Hiperbolik Tanjant (tanh) Fonksiyonu

YSA modellerinde kullanılan hiperbolik tanjant (tanh) fonksiyonu sigmoid fonksiyonuna oldukça benzeyen ve sınıflandırma için kullanılan fonksiyondur. Aktivasyon fonksiyon çıkış değeri [-1,1] aralığında doğrusal olmayan bir fonksiyondur.

1.4.3. Rectified Linear Unit / Rektifiye Doğrusal Birim (ReLu) Fonksiyonu

YSA modellerinde ReLU aktivasyon fonksiyonu çok gizli katmanlarda sıklıkla kullanılıp aynı anda tüm nöronları aktive etmez. ReLU aktivasyon fonksiyonunda negatif değerler üreten nöronlar sıfır değeri kabul edilir. Böylece ReLU aktivasyon fonksiyonu modeli daha verimli ve hızlı eğitir.

1.4.4. Leaky ReLu Fonksiyonu

YSA modellerinde Leaky ReLU aktivasyon fonksiyonu ReLU fonksiyonunda benzer biçimdedir. Ancak Leaky ReLU fonksiyonunda negatif değerler ReLU fonksiyonunda olduğu gibi tam sıfır yerine sıfıra yakın negatif bir değer alır. Böylece öğrenmenin çift yönlü olarak da gerçekleşmesi sağlayan bir aktivasyon fonksiyonudur.

1.4.5. Maxout Fonksiyonu

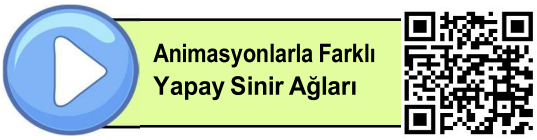
Maxout fonksiyonu öğrenilebilen, hesaplamalı olarak ucuz ve yalnızca etken giriş parametresini dikkate alan YSA modellerinde kullanılan bir aktivasyon fonksiyonudur. Böylece en etkili giriş parametresi seçilerek elde edilen sonuçlar çıkışa gönderir.

1.4.6. Üst Lineer Birim (Exponential Linear Unit)-ELU Fonksiyonu

YSA modellerinde kullanılan ELU fonksiyonu negatif girdiler hariç ReLU fonksiyonuna oldukça benzerdir. ELU fonksiyonunda orta nokta "0" sıfırdır. Böylece ölü nöronların önüne geçerek modelin daha hızlı yakınsaması sağlanabilir.

1.5. Yapay Sinir Ağları Kullanım Alanları

Günümüzde bilgisayarlar ve bilgisayar sistemleri yaşamımızın vazgeçilmez bir parçası haline gelmiştir. Yapay zekânın alt dallarından birisi olan YSA, bilgisayarların derin öğrenme yöntemlerinde sıklıkla kullanılan bir yapıdır. YSA temel alınarak farklı birçok derin öğrenme mimarisi tasarlanmıştır. YSA modelleri sağlık, medikal ve tıp sektörleri başta olmak üzere makine imalat sanayisi, otomotiv, elektronik, havacılık ve uzay sanayisi, bankacılık ve askeri alanlarda etkili biçimde kullanılmaktadır. Ayrıca insansı robotlarda sıklıkla kullanılmaktadır (Kaya, 2018).



Eğitmene Not

Eğitmen, '*Animasyonlarla Farklı Yapay Sinir Ağları*' videosunu öğrencilere izleterek, sinir hücresinin farklı ağ modelleri geliştirmede nasıl araç olabileceğini vurgular; insan beyni ve hatta iletişim ağları (bilgisayar ağları, sosyal ağlar vs.) üzerinden yola çıkarak iş birliği içerisindeki bilgi aktarımının Yapay Zekâ'daki öğrenme süreçleri için ne kadar önemli bir esin kaynağı olduğunu açıklar.

2. TASARLA

Eğitmene Not

Eğitmen, öğrencilere yapay sinir ağları ile pamuk tarlasına ait sıcaklık ve nem giriş parametrelerine göre akıllı sulama sistemi ile ilgili şu ifadeleri aktarır:

"*Günümüzde su kaynaklarının gün geçtikçe azalması sebebiyle tarım sektöründe kullanılan suların daha verimli bir şekilde kullanılması oldukça önemlidir.* ***Yapay sinir ağları*** *ile oluşturulan gerçek zamanlı kontrol ile çiftçilerin ekinlerine otomatik su temini mümkün olmaktadır. Bu durum hem ürün kalitesinin artmasına hem de fazla miktarda su kullanılmamasına sebep olmaktadır. Akıllı sulama sistemi ile eşik ve gerekli su seviyeleri sağlanarak su israfı önlenir.* ***Yapay sinir ağlarının*** *temel amacı, toprağın nemine ve ortam sıcaklık ölçümünden elde edilen verilere göre pompa motorunun açıp kapalma kararının alınmasına yardımcı olan akıllı bir sulama sistemi oluşturmaktır. Dolayısıyla* ***yapay sinir ağları*** *ile akıllı sulama sistemi sayesinde su ve çevre korunmaktadır.*"

Bu bölümde pamuk tarlası uygulamasına ait 200 adet veriden oluşan bir veri setinden faydalanılarak (Web Kaynağı 5.1), sıcaklık ve nem (%rh) giriş parametrelerine göre pamuk tarlasının sulama pompa motorunun çalışıp çalışmaması durumunun yapay sinir ağları ile tahminlenmesi amaçlanmıştır. Öğrenciler, ilk olarak yapay sinir ağları ile pamuk tarlasına ait

sıcaklık ve nem giriş parametrelerine göre pompa ile akıllı sulama yapılıp yapılmayacağının değerlendirilmesi problemini tartışır. Şekil 5.6'da sulama görüntüsü verilmiştir.

Şekil 5.6. Sulama sistemi temsili (Web Kaynağı 5.2).

Öğrenciler veri setinden de gözlemleyebilecekleri, Çizelge 5.1'de örnek olarak gösterilen veri setine karşı, yapay sinir ağları tabanlı sulama tespit çözümü için giriş çıkış parametrelerini irdeler.

Çizelge 5.1. Yapay sinir ağları için giriş ve çıkış parametreleri

Veri Sayısı	GİRİŞ PARAMETRELERİ		ÇIKIŞ PARAMETRESİ
	Sıcaklık	Nem (%rh)	Pompa Motor
1	16	62,36	1 (Çalışıyor)
2	18	51,02	1
3	22	72,43	1
4	32	78	1
5	28	67,44	1
6	23	54,54	1
...	...	...	...
102	42	8,69	0 (Çalışmıyor)
103	35	7,42	0
104	19	42,42	0
105	16	47,60	0
...	...	...	...
198	42	87,38	1
199	45	99,90	1
200	10	95,69	1

VERİ SETİ İÇİN İNDİRME LİNKİ

https://github.com/deneyapyz/ortaokul/blob/main/Hafta5/INTELLIGENT%20IRRIGATION%20SYSTEM.csv

Eğitmene Not

Eğitmen, öğrencilerden ilgili indirme linkini kullanarak, orijinal veri setinin (https://www.kaggle.com/harshilpatel355/autoirrigationdata/download), bu uygulamada kullanımına uygun; dönüştürülmüş versiyonunu indirmelerini ister.

Düşün, tartış…

Problem çözümünde yapay sinir ağları kullanmanın avantajları neler olabilir? Diğer makine öğrenme algoritmaları tercih edilse sonuçlar daha iyi olabilir mi? Tartışalım…

Super Mario Oynamasını Öğrenen Yapay Zekâ!

3. HAREKETE GEÇ

Eğitmene Not

Eğitmen, öğrencilere ilk önce kod satırlarını tek tek yazdırarak her kod satırında öğrenciye kod satırının ne anlama geldiği ile ilgili sorular sorar ve tartışır.

Şekil 5.7'de gösterildiği gibi "INTELLIGENT IRRIGATION SYSTEM.csv" dosyasını yüklemek için gerekli kütüphaneyi yazar (Söz konusu kodun tamamı ilgili Github platformunda Hafta5 klasörü altındaki H5_ysa_sulama.py dosyasında paylaşım durumundadır).

1 nolu kod satırında yapay zekâ kütüphanesi olan **Keras** kütüphanesinin **Sequential** fonksiyonu ile boş bir yapay sinir ağı modeli oluşturmak için gerekli olan sınıf çağrılır.

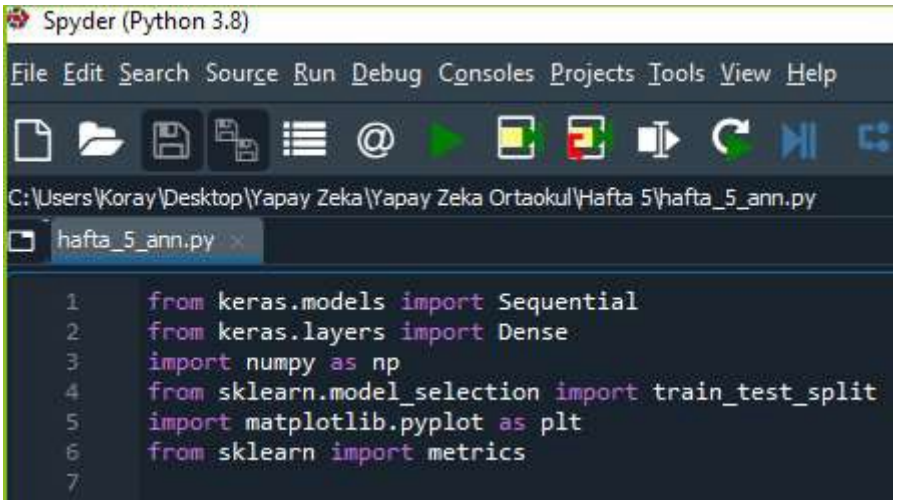
2 nolu kod satırında yapay zekâ kütüphanesi olan **Keras** kütüphanesinin **Dense** fonksiyonu ile yapay sinir ağı tam bağlı katmanlarını tanımlamak için gerekli olan sınıf çağrılır.

3 nolu kod satırında matematiksel dizi işlemlerini gerçekleştirmek için **Numpy** kütüphanesi çağrılır.

4 nolu kod satırında veri setini bölmek için **sklearn.model_selection** kütüphanesinden **train_test_split** fonksiyonu çağrılır.

5 nolu kod satırında model eğitiminden elde edilen sonuçları çizdirmek için **matplotlib.pyplot** kütüphanesi çağrılır.

6 nolu kod satırında ise **sklearn** kütüphanesinden **metrics** fonksiyonu ile modelin değerlendirebilmesi için gerekli fonksiyon çağrılır.



Şekil 5.7. Gerekli kütüphanelerin kod gösterimi

Öğrenciler Şekil 5.8'deki veri seti ile ilgili düzenlemeleri 8-11 kod satırları arasında gerçekleştirir.

8. kod satırında numpy kütüphanesinin **genfromtxt** özelliğini kullanarak veri seti ile aynı konumda olan kod dosyasının içerisindeki INTELLIGENT IRRIGATION SYSTEM.csv veri dosyasını **","** ayracına göre (delimiter) okuyarak dataset değişkenine aktarır.

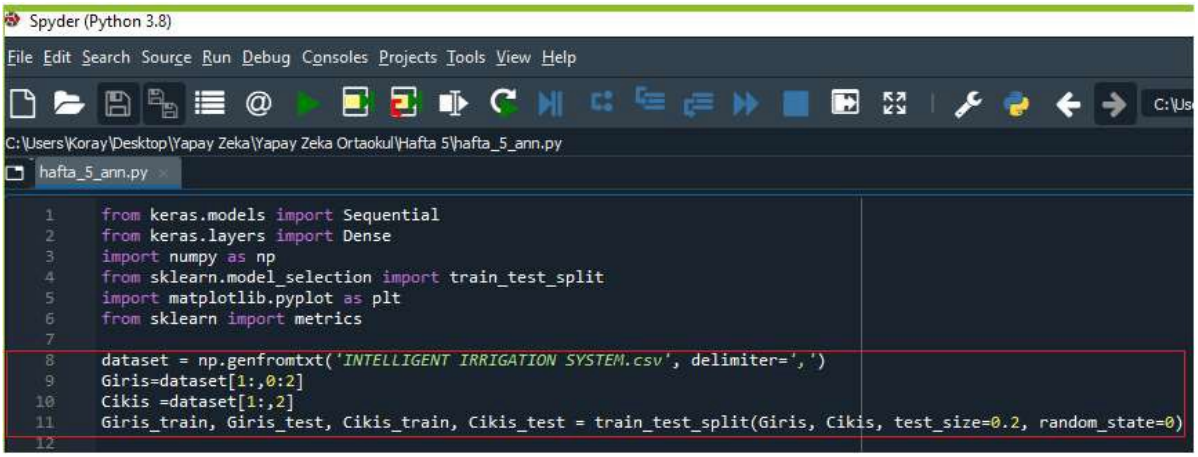
9. kod satırında dataset değişkenindeki veri setindeki ilk iki sütunu ([1:,0:2]) giris parametresi olarak ayarlayarak "**Giris"** değişkenine aktarır.

Eğitmene Not

Eğitmen, kod satırında yer alan ([1:,0:2]) ifadesine açıklık getirerek; virgülden önce yer alan **"1:"** yazım şeklinin, veri setindeki başlık satırının (Nem, Sıcaklık ve Pompa) eğitim sürecine katılmaması içinyazıldığını belirtir. Yine **"0:2"** yazım şeklinin de veri setinde ilk iki sütun olan; nem ve sıcaklık değerlerinin giriş parametresi olarak ayarlanması için kullanıldığını ifade eder.

10. kod satırında dataset değişkenindeki veri setindeki üçüncü sütunu ([1:, 2]) çıkış parametresi olarak ayarlayarak "**Cikis"** değişkenine aktarır.

11. kod satırında veri setindeki eğitim ve test verilerini %80 eğitim, %20 test olacak şekilde **"test_size=0,2"** komutu kullanarak rastgele ayırır.



Şekil 5.8. Veri setinin giriş, çıkış ve eğitim için bölünmesi kod ekranı

Öğrenciler 13-19 kod satırları arasında veri setinin eğitimi için yapay sinir ağları modelini kurar.

13. kod satırında "**Sequential"** fonksiyonu ile yeni bir boş yapay sinir ağı modeli oluşturarak YSA modelinin ismini **"model"** değişkenine aktarır.

14. kod satırında oluşturulan YSA modeli "add (Dense)" kodu ile 6 nöronlu gizli katman, 2 girişli ("input_dim") ve Relu aktivasyon fonksiyonu ilk katmanı oluşturur.

15. kod satırında 6 nöronlu gizli katman ve Relu aktivasyon fonksiyonlu ikinci katmanı oluşturur.

16. kod satırında 6 nöronlu gizli katman ve Relu aktivasyon fonksiyonlu üçüncü katmanı oluşturur.

17. kod satırında 6 nöronlu gizli katman ve Relu aktivasyon fonksiyonlu dördüncü katmanı oluşturur.

18. kod satırında 1 nöronlu çıkış katmanı ve sigmoid aktivasyon fonksiyonlu son katmanı oluşturur.

19. kod satırında **"compile"** fonksiyonu ile kayıp veri değerlerini **"binary_crossentropy",** optimizasyon yöntemi olarak **"adam"** ve ölçüm metriği olarak **"accuracy"** parametrelerine göre derler.

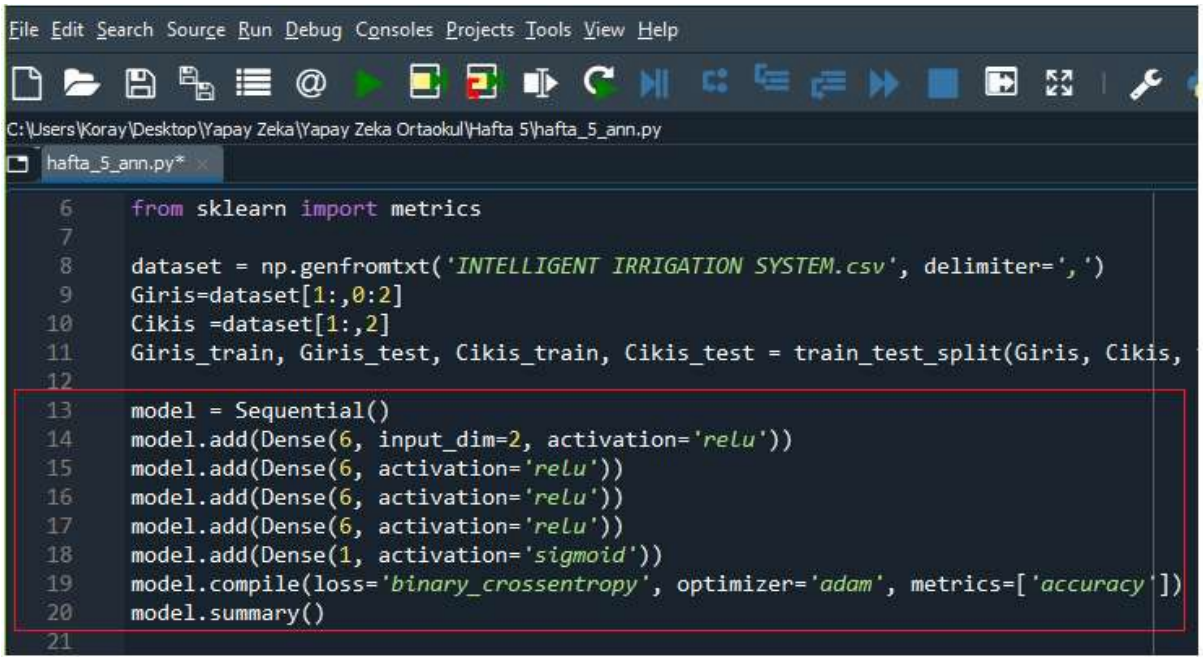
Eğitmene Not

Eğitmen, öğrencilere kod satırında yer alan **"binary_crossentropy**", ifadesini kayıp fonksiyonu iki sınıflı problemlerde kullanıldığını açıklar. Örneğin çalışmada kullanılan veri setinde pompa motorunun açık veya kapalı olması iki sınıflı bir problemdir. Bu aşamada devreye bu kayıp fonksiyonu girmektedir. Örneğin pompa motor çalışıyor ise 1, çalışmıyorsa 0 olarak işaretleyebilir.

Şekil 5.9'da gösterildiği gibi 20. kod satırında oluşturulan yapay sinir ağı modelindeki nöronların, giriş çıkış parametrelerine ait özellikleri **".summary"** fonksiyonu ile görür.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_10 (Dense)             (None, 6)                 18
_________________________________________________________________
dense_11 (Dense)             (None, 6)                 42
_________________________________________________________________
dense_12 (Dense)             (None, 6)                 42
_________________________________________________________________
dense_13 (Dense)             (None, 6)                 42
_________________________________________________________________
dense_14 (Dense)             (None, 1)                 7
=================================================================
Total params: 151
Trainable params: 151
Non-trainable params: 0
_________________________________________________________________
```

Şekil 5.9. YSA modelin özeti



Şekil 5.10. Yapay sinir ağı modelinin kurulum için kod ekranı

4. YÜRÜT

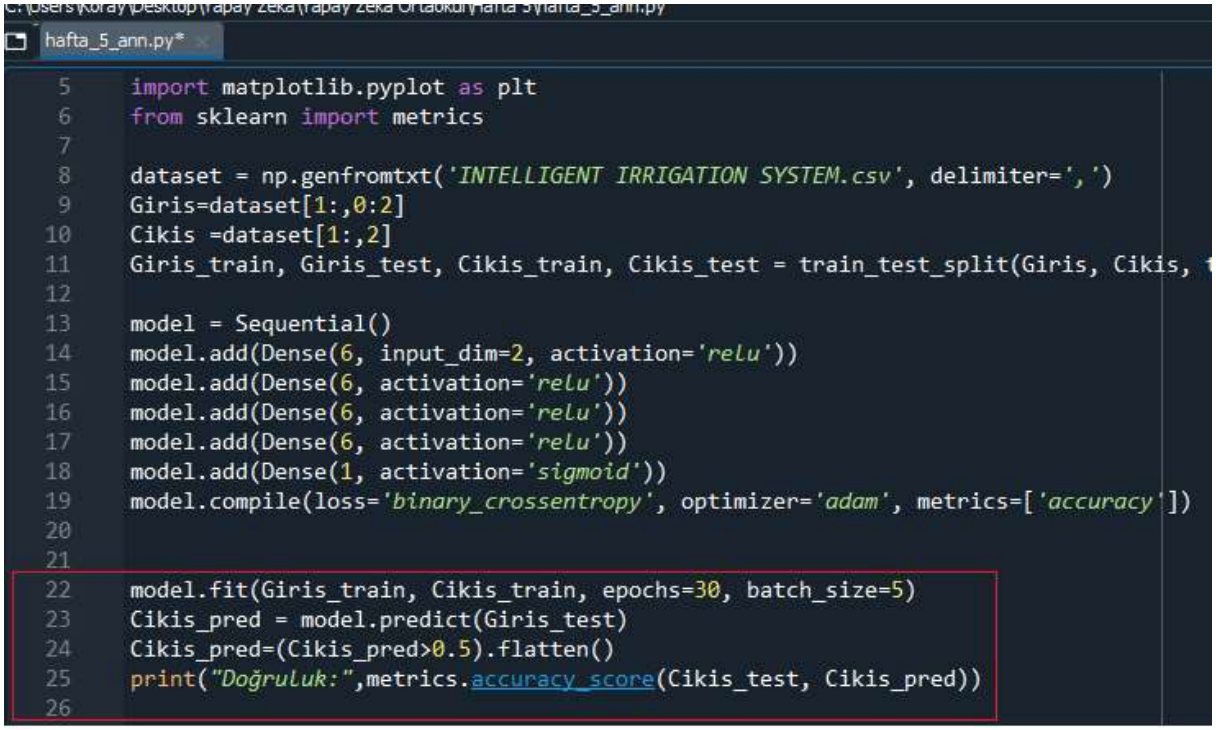
Öğrenciler 22-25 satırları arasında pamuk tarlası için kurulan yapay sinir ağı modelinin eğitimini gerçekleştirir.

22. kod satırında **"model.fit"** komutu yazarak giriş eğitim verilerine göre çıkış eğitim verileri için Yapay sinir ağları ile eğitim gerçekleştirir. Burada her bir eğitim için **"batch_size"** fonksiyonuna aktarılan 5 değeri ile eğitime alınacak veri sayısı belirlenerek 30 adet eğitim (**epochs=30**) gerçekleştirilir.

23. kod satırında ise YSA ile eğitim işlemi sonunda giriş test verilerine göre **".predict"** fonksiyonu ile pompa motorunun çalışıp çalışmaması durumu için tahminini gerçekleştirir.

24. kod satırında YSA modeli sınıflandırma işleminde kullanılan **"sigmoid"** fonksiyonundan dolayı (sigmoid fonksiyonu 0 ile 1 arasında lineer olmayan değerler üretir) model elde edilen sonuçlar 0,5'ten büyük ise **"1"** sınıfına, 0,5'den küçük ise **"0"** sınıfına aktarılarak ikili sınıflandırma yapılmıştır. Elde edilen iki boyutlu dizi sonuçları **".flatten"** fonksiyonu kullanılarak tek boyutlu diziye çevrilmiştir.

25. kod satırında YSA modelinden elde edilen tahmin sonuçları ile gerçek sonuçlar karşılaştırılarak bulunun doğruluk değeri **"print"** komutu kullanılarak konsola yazdırılır.



Şekil 5.11. Yapay sinir ağları modelinde eğitim ve tahminleme işlemleri için kod ekranı

5. KARAR VER

5.1. Tahmin-Test Sonuçlarını Karşılaştırma

Öğrenciler, YSA modeli ile eğitilen pamuk tarlası sulama durumu tahminlemesine ait hata matrisini (confusion matrix) kullanarak modelin başarısını ölçer. Şekil 5.12 gösterildiği gibi 29 numaralı komut satırında, **"sklearn.metrics"** kütüphanesinden **"confusion_matrix"** özelliği yüklenir.

30. numaralı satırda ise, hata matrisini görüntüleyebilmek için gerekli olan **"seaborn"** kütüphanesini yükler.

31. numaralı satırda ise, hata matrisindeki grafikleri çizebilmek için **matplotlib** kütüphanesi içerisinde yer alan **pyplot** fonksiyonunu yükler.

32. numaralı kod satırında YSA modelden elde edilen veri seti üzerindeki işlemleri yapabilmek için gerekli olan **"pandas"** kütüphanesini yükler.

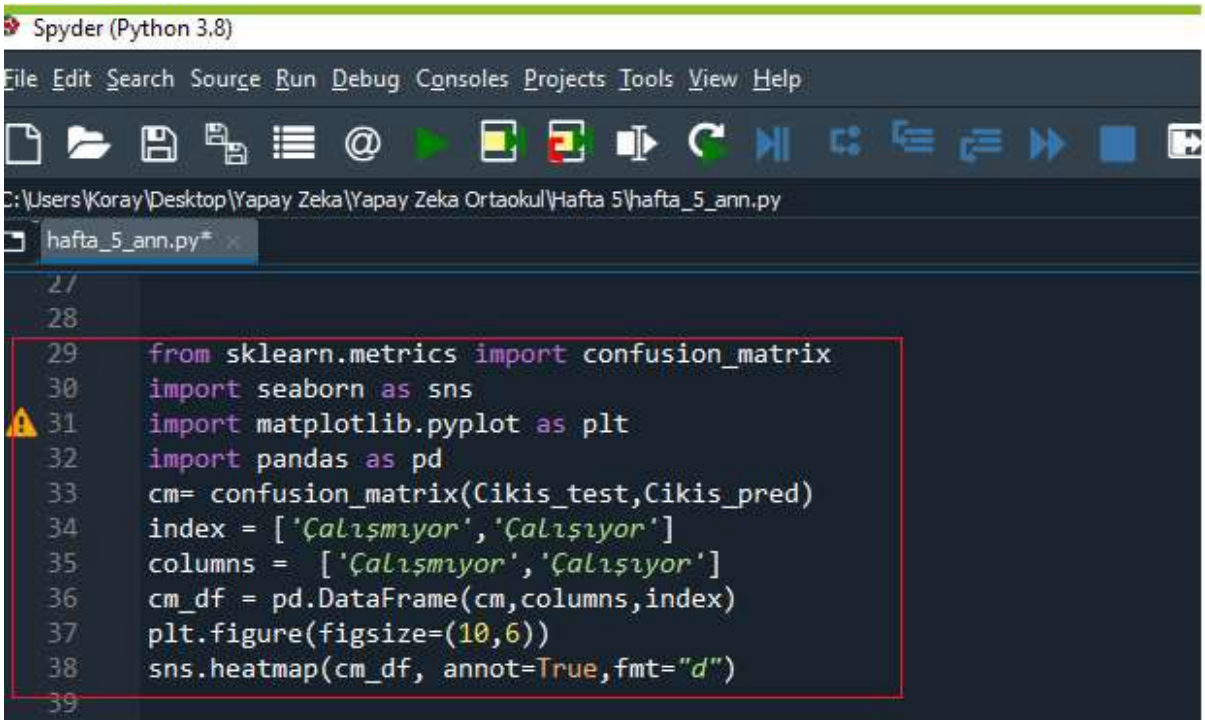
33. numaralı kod satırında ise, cm değişkeni ile oluşturulan hata matrisinin satır (Cikis_test) ve sütunlarını (Cikis_tahmin) oluşturur.

34. ve 35. numaralı kod satırlarında ise index ve colums değişkenleri kullanarak hata matrisinde yer alacak olan metinleri belirler.

36. numaralı kod satırında **cm_df** değişkeni ile Cikis_test ve Cikis_tahmin değerlerinin veri çerçevesine (DataFrame) aktarılmasını sağlar.

37. numaralı kod satırında ise **"plt.figure"** komutu ile 10x6 cm çerçeve boyutunda boş bir çizim ekranı açar.

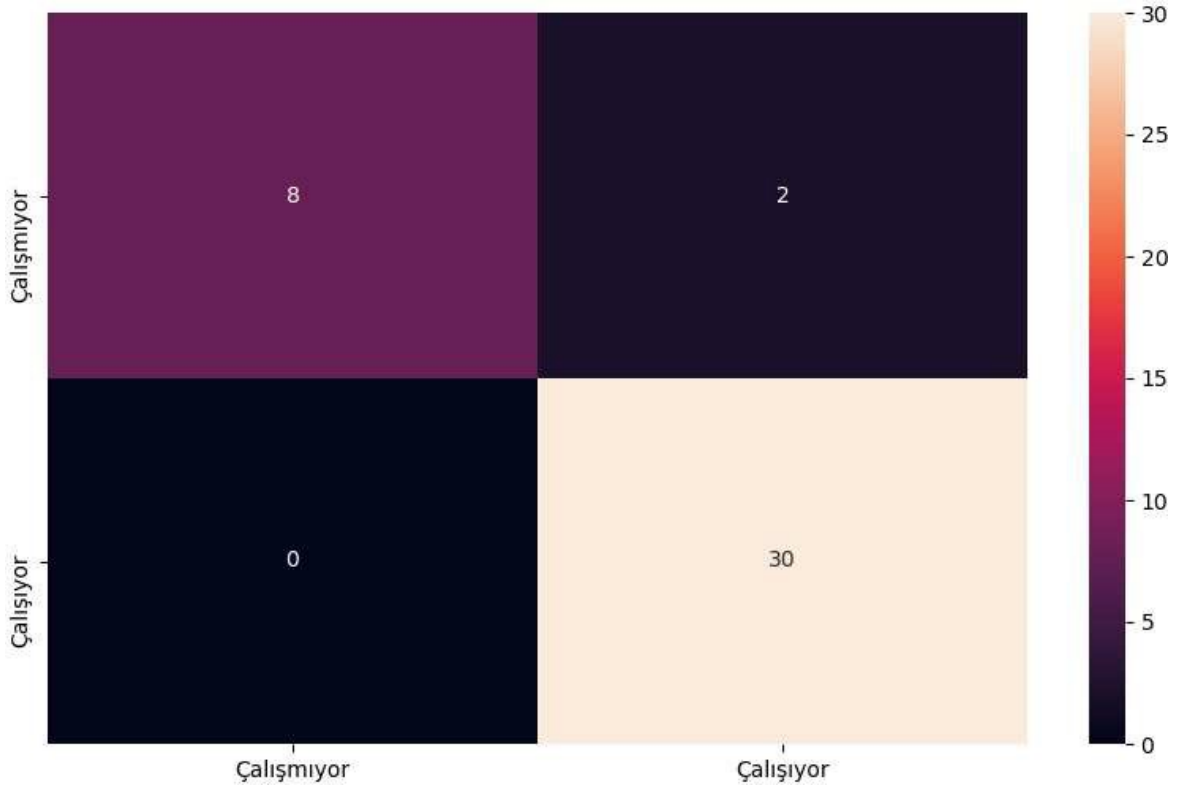
38. numaralı kod satırında ise **"sns.heatmap** komutu ile oluşturulan veri çerçevesini renkli olarak çizer. Burada annot=True ile sayısal değerler gösterilirken, fmt="d" ile sayısal değerler tam sayı olarak gösterilir.



Şekil 5.12. YSA modelinde eğitim ve tahminleme işlemleri için kod ekranı

5.2. Hata Matrisini Değerlendirme

Öğrenciler, pamuk tarlası sulama durumuna ait toplam 200 adet veri setinde (Web Kaynağı 5.1) kayıtlı verinin %80'ini (160) eğitim, %20'i (40) test eğitim seti olarak ayırmıştı. YSA modeli 160 kayıt ile eğitmişti. Geriye kalan 40 kayıt (Giris_test) ise modeli tahmin etmek için kullandı (Cikis_tahmin). Öğrenciler, Şekil 5.13'te gösterildiği gibi 40 test kaydına ait gerçek sonuçlar (Cikis_test) ile tahmin sonuçlarını karşılaştırdı ve hata matrisi elde etti. Böylece, matristeki sayılar toplamının test verisi olan 40'a eşit olduğu gördü.



Şekil 5.13. Hata matrisi

Eğitmene Not

Eğitmen, öğrenciler ile hata matrisini inceler. Öğrencilere hata matrisindeki değerlerin ne anlama geldiğini sorar ve şu yönde cevapları öğrencilerden geri bildirim olarak almaya çalışır:

"Hata matrisi incelendiğinde, 40 adet test kaydında pamuk tarlasında pompa motorun çalışıp çalışmama durumuna ait 10 tane çalışmıyor test verisinden 8 adetinin doğru (çalışmıyor), 2 adetinin ise yanlış (çalışıyor) tahmin edildiği görülmüştür. 30 adet çalışıyor test verisinden 30 adetin doğru (çalışıyor), 0 adetin yanlış (çalışmıyor) tahminlediği görülmüştür (Öğrenciler Şekil 5.14'te gösterildiği gibi YSA modeli ile pamuk tarlası sulama sistemi için konsol ekranında %95 başarım oranında tahminlendiğini görür.)."

```
Epoch 29/30
32/32 [==============================] - 0s 1ms/step - loss: 0.2009 - accuracy: 0.9047
Epoch 30/30
32/32 [==============================] - 0s 1ms/step - loss: 0.1578 - accuracy: 0.9403
2/2 [==============================] - 0s 0s/step - loss: 0.1504 - accuracy: 0.9500

Doğruluk: 0.95
```

Şekil 5.14. YSA modeli ile pamuk tarlası sulama sistemi için doğruluk sonucu

Eğitmene Not

Eğitmen, öğrenciler ile 30 Epoch sayısı ile gerçekleştirilen eğitimdeki kayıp (loss) ve doğruluk (accuracy) elde edilen sonuçlar ile test verileri ile modelden elde edilen sonuçları tartışır ve açıklar.

PYTHON KODLARI:

```
from keras.models import Sequential
from keras.layers import Dense
import numpy as np
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn import metrics

dataset = np.genfromtxt('INTELLIGENT IRRIGATION SYSTEM.csv', delimiter=',')
Giris=dataset[1:,0:2]
Cikis =dataset[1:,2]
Giris_train, Giris_test, Cikis_train, Cikis_test = train_test_split(Giris, Cikis,
test_size=0.2, random_state=0)

model = Sequential()
model.add(Dense(6, input_dim=2, activation='relu'))
model.add(Dense(6, activation='relu'))
model.add(Dense(6, activation='relu'))
model.add(Dense(6, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])
model.summary()
```

```
model.fit(Giris_train, Cikis_train, epochs=30, batch_size=5)
Cikis_pred = model.predict(Giris_test)
Cikis_pred=(Cikis_pred>0.5).flatten()
print("Doğruluk:",metrics.accuracy_score(Cikis_test, Cikis_pred))

from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
cm= confusion_matrix(Cikis_test,Cikis_pred)
index = ['Çalışmıyor','Çalışıyor']
columns =  ['Çalışmıyor','Çalışıyor']
cm_df = pd.DataFrame(cm,columns,index)
plt.figure(figsize=(10,6))
sns.heatmap(cm_df, annot=True,fmt="d")
```

Düşün, tartış…

Bir Yapay Sinir Ağları'nın farklı verilerle karşılaşarak yanlış öğrenme gerçekleştirmesini sağlamak mümkün olabilir mi? Tartışalım…

Dünya'dan Haberler

Türkiye'nin Ulusal Yapay Zekâ Stratejisi

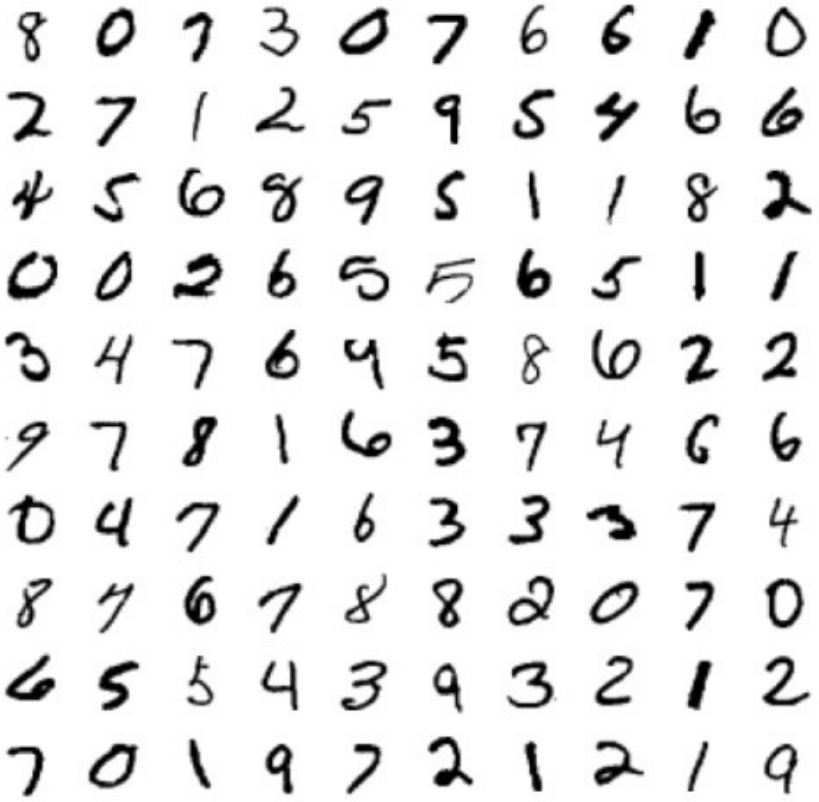
Cumhurbaşkanlığı Dijital Dönüşüm Ofisi Başkanlığı ile Sanayi ve Teknoloji Bakanlığı iş birliğinde ve ilgili tüm paydaşların etkin katılımıyla hazırlanan "*Ulusal Yapay Zekâ Stratejisi 2021-2025*"e ilişkin 2021/18 sayılı Cumhurbaşkanlığı Genelgesi 20/08/2021 tarihli ve 31574 sayılı Resmî Gazete'de yayımlanarak yürürlüğe girmiştir.

Yapay zekâ alanında ülkemizin ilk ulusal strateji belgesi olma özelliğini taşıyan Ulusal Yapay Zekâ Stratejisi (UYZS) ile Türkiye, yapay zekâ (YZ) stratejisini yayımlayan ülkeler arasında yerini almıştır. UYZS, On Birinci Kalkınma Planı ile Cumhurbaşkanlığı Yıllık Programları doğrultusunda, "Dijital Türkiye" vizyonu ve "Milli Teknoloji Hamlesi" ile uyumlu olarak hazırlanmıştır. Vizyonu "*müreffeh bir Türkiye için çevik ve sürdürülebilir yapay zekâ ekosistemiyle küresel ölçekte değer üretmek*" olan Strateji, 6 stratejik öncelik etrafında tasarlanmıştır. (1) YZ Uzmanlarını Yetiştirmek ve Alanda İstihdamı Artırmak, (2) Araştırma, Girişimcilik ve Yenilikçiliği Desteklemek, (3) Kaliteli Veriye ve Teknik Altyapıya Erişim İmkânlarını Genişletmek, (4) Sosyoekonomik Uyumu Hızlandıracak Düzenlemeleri Yapmak, (5) Uluslararası İş Birliklerini Güçlendirmek, (6) Yapısal ve İşgücü Dönüşümünü Hızlandırmak.

UYZS ile 2021-2025 yılları arasında ülkemizde yürütülen YZ alanındaki çalışmaları ortak bir zemine oturtacak tedbirler ve bu tedbirleri hayata geçirmek üzere oluşturulacak yönetişim mekanizması ortaya konulmaktadır (Web Kaynağı 5.3).

6. İLAVE ETKİNLİK

Şekil 5.15'te gösterildiği gibi MNIST (Modified National Institue of Standarts) veri seti içerisinde yer alan 60000 adet 28x28 piksel boyutuna sahip farklı insanların el yazmalarından kesilmiş resimlerden oluşmaktadır. Bunlara karşılık gelen 60000 tane sayı (skaler) vardır. MNIST veri setini yapay sinir ağları kullanarak veri seti içerisinde yer alan resmin hangi rakama ait olduğunu tahminleyen bir model oluşturunuz (Veri seti keras kütüphanesi içerisinden otomatik çekilmektedir. İlgili kodun tamamı Github platformunda, Hafta 5 altında H5_ysa_mnist.py dosyası olarak paylaşılmış durumdadır.).



Şekil 5.15. El yazısı görüntüsü

Eğitmene Not

Eğitmen, öğrencilere Python içinde vektörize edilmiş bir biçimde, yani resimlerin piksel değerleriyle sayısallaştırılmış halinin hazır bulunduğunu açıklar. Çizelge 5.2'de gösterildiği gibi 28x28 boyutundaki el yazması görüntüler giriş parametresini, bu görüntülere ait sayısallaştırmış değeri çıkış sınıfını temsil ettiğini anlatır.

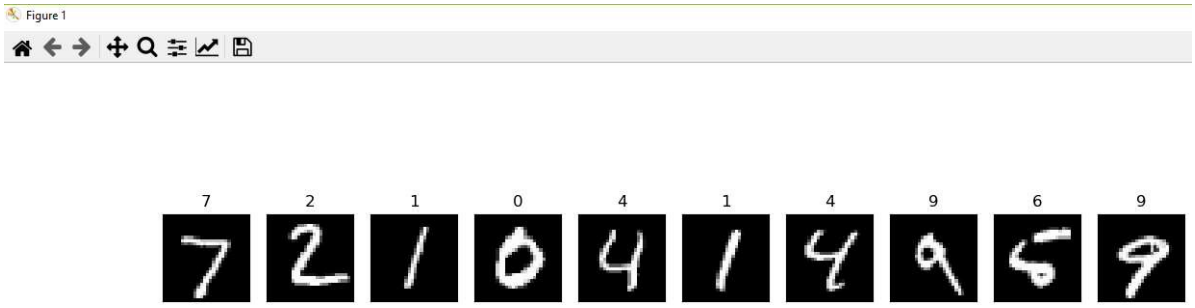
Çizelge 5.2. Veri setinin giriş çıkış parametreleri

Veri Sayısı	GİRİŞ PARAMETRESİ	ÇIKIŞ PARAMETRESİ
	Görüntü	Sınıf
1		8
2		0
...	...	...
59000		7
60000		6

PYTHON KODLARI:

```
import numpy as np
import matplotlib.pyplot as plt
from keras.layers import Dense, Flatten
from keras.models import Sequential
from tensorflow.keras.utils import to_categorical
from keras.datasets import mnist

(X_train, y_train), (X_test, y_test) = mnist.load_data()
print(X_train.shape)
print(X_test.shape)

temp = []

for i in range(len(y_train)):
    temp.append(to_categorical(y_train[i], num_classes=10))
y_train = np.array(temp)
# Convert y_test into one-hot format
temp = []
for i in range(len(y_test)):
    temp.append(to_categorical(y_test[i], num_classes=10))
y_test = np.array(temp)

model = Sequential()
model.add(Flatten(input_shape=(28,28)))
model.add(Dense(16, activation='sigmoid'))
model.add(Dense(10, activation='softmax'))
model.summary()

model.compile(loss='categorical_crossentropy',
```

```
            optimizer='adam',
            metrics=['acc'])

model.fit(X_train, y_train, epochs=15,
          validation_data=(X_test,y_test))

predictions = model.predict(X_test)
predictions = np.argmax(predictions, axis=1)

fig, axes = plt.subplots(ncols=10, sharex=False,
                         sharey=True, figsize=(20, 4))
for i in range(10):
    axes[i].set_title(predictions[i])
    axes[i].imshow(X_test[i], cmap='gray')
    axes[i].get_xaxis().set_visible(False)
    axes[i].get_yaxis().set_visible(False)
plt.show()
```



Şekil 5.16. El yazısı genel tanımlama görüntüleri

```
Epoch 1/15
1875/1875 [==============================] - 4s 2ms/step - loss: 1.0922 - acc: 0.6976 - val_loss: 0.6564 - val_acc: 0.8255
Epoch 2/15
1875/1875 [==============================] - 3s 2ms/step - loss: 0.5876 - acc: 0.8316 - val_loss: 0.5526 - val_acc: 0.8375
Epoch 3/15
1875/1875 [==============================] - 3s 2ms/step - loss: 0.5009 - acc: 0.8514 - val_loss: 0.4441 - val_acc: 0.8717
Epoch 4/15
1875/1875 [==============================] - 3s 2ms/step - loss: 0.4859 - acc: 0.8538 - val_loss: 0.4770 - val_acc: 0.8582
Epoch 5/15
1875/1875 [==============================] - 3s 2ms/step - loss: 0.4550 - acc: 0.8642 - val_loss: 0.4353 - val_acc: 0.8689
Epoch 6/15
1875/1875 [==============================] - 3s 2ms/step - loss: 0.4414 - acc: 0.8662 - val_loss: 0.4198 - val_acc: 0.8703
Epoch 7/15
1875/1875 [==============================] - 3s 2ms/step - loss: 0.4269 - acc: 0.8720 - val_loss: 0.4199 - val_acc: 0.8782
Epoch 8/15
1875/1875 [==============================] - 3s 2ms/step - loss: 0.4331 - acc: 0.8705 - val_loss: 0.3994 - val_acc: 0.8855
Epoch 9/15
1875/1875 [==============================] - 3s 2ms/step - loss: 0.4315 - acc: 0.8734 - val_loss: 0.4424 - val_acc: 0.8638
Epoch 10/15
1875/1875 [==============================] - 4s 2ms/step - loss: 0.4144 - acc: 0.8763 - val_loss: 0.4027 - val_acc: 0.8851
Epoch 11/15
1875/1875 [==============================] - 5s 3ms/step - loss: 0.3985 - acc: 0.8798 - val_loss: 0.3734 - val_acc: 0.8877
Epoch 12/15
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4034 - acc: 0.8796 - val_loss: 0.4140 - val_acc: 0.8717
Epoch 13/15
1875/1875 [==============================] - 5s 3ms/step - loss: 0.3992 - acc: 0.8813 - val_loss: 0.3721 - val_acc: 0.8944
Epoch 14/15
1875/1875 [==============================] - 5s 3ms/step - loss: 0.3813 - acc: 0.8880 - val_loss: 0.3669 - val_acc: 0.8968
Epoch 15/15
1875/1875 [==============================] - 4s 2ms/step - loss: 0.3783 - acc: 0.8887 - val_loss: 0.3883 - val_acc: 0.8897
```

Şekil 5.17. Epoch bazlı genel akış

Eğitmene Not

Eğitmen, ilgili hafta eğitimi hakkında öğrencilerin ilgisini arttırmak ve yeni öğrenilen bilgileri önceden öğrenilen bilgiler ile karşılaştırmak için aşağıda maddeler halinde verilen soruları tartışır.

- İlgili haftada ele alınan yapay zekâ uygulamalarına uygun örnek veri setleri neler olabilir?
- İlgili haftada ele alınan veri setlerinin yapay zekâ teknik ve kütüphanelerinde ne derecede başarım etkisi olmuştur?
- İlgili haftada öğretilen yapay zekâ tekniği ile uyumlu ve uyumsuz olabilecek açık erişimli veri seti çeşitleri sunan (kaagle, github vs.) internet sitelerinden elde edilecek farklı veri çeşitleri ile yapay zekâ tekniği uyumluluğunu sorar.

.

Kaynakça

Kaya, Ü., Oğuz, Y., ve Şenol, Ü. (2018). An Assessment of Energy Production Capacity of Amasra Town Using Artificial Neural Networks. Turkish Journal of Electromechanics and Energy, 3(1), 22-26.

Web Kaynağı 5.1: https://www.kaggle.com/harshilpatel355/autoirrigationdata/download

Web Kaynağı 5.2: https://gremonsystems.com/blog-en/things-you-didnt-know-about-automatic-watering-systems/

Web Kaynağı 5.3: https://cbddo.gov.tr/uyzs

6. Hafta: Etmen Tabanlı Modelleme

Ön Bilgi:

- Etmen tabanlı Yapay Zekâ kavramının temelleri, etmen tabanlı modellemenin temelleri
- Python kodlamada döngüler, fonksiyonlar ve kütüphane yapılarının kullanımı (Bu konuda Python anlatım videoları: 13. Fonksiyonlar, 14. Python Kütüphane Kullanımı yardımcı araç olarak kullanılabilecektir).

Haftanın Kazanımları:

- Öğrenciler etmen tabanlı modelleme kavramını kavrar.
- Öğrenciler, etmen modelleme üzerine problem çözümleri tasarlayarak uygular.
- Öğrenciler etmen tabanlı modellemenin çözüm süreçlerini yorumlamayı kavrar.
- Öğrenciler birbirleriyle etkileşen etmenlerle problem çözümü üretme yeteneği kazanır.
- Öğrenciler, etmen, çevre, dönüt, çoklu-etmen gibi kavramları anlar.
- Öğrenciler Python programlama dilini kullanarak tipik bir etmen tabanlı çözüme yönelik program kodunu yazabilir.
- Öğrenciler Python programlama dilinde etmen tabanlı modelleme için kullanılan kütüphane fonksiyonlarını kullanma yeteneği kazanır.
- Öğrenciler, örnek bir probleme göre etmen tabanlı modelleme yeteneği kazanır.
- Öğrenciler etmen tabanlı çözüm gerektiren problem için model tasarlar ve bu modeli çözüm yönünde uygular.
- Öğrenciler modelin davranışlarını değerlendirerek yorumlar.
- Öğrenciler etmen modelli programlamanın Yapay Zekâ alanı açısından önemini kavrar.
- Bir sonraki zeki optimizasyon konusuna temel olan bilgiyi (etmen mantığı) kazanır.

Haftanın Amacı:

Bu haftanın amacı, “etmen (agent / ajan)” kavramının tüm öğrenciler tarafından doğru bir şekilde anlaşılmasını sağlamaktır. Haftanın bir diğer amacı, etmen tabanlı modelleme yaklaşımını izleyerek alternatif problem yapıları ile öğrencilerin ilgisini çekecek etkinlikleri gerçekleştirmektir. Yine bu bağlamda, öğrencilerin Python programlama dilini kullanarak etmen tabanlı örnek problem modelleme ve çözüm üretme konusunda yetenek kazanması sağlanacaktır.

Kullanılacak Malzemeler:

Bilgisayar, kâğıt, kalem, Python kod editörü, örnekler için çıktı görselleri

Haftanın İşlenişi:

Algıla: Etmen ve etmen tabanlı modelleme kavramlarını öğrenir.

Tasarla: Etmen tabanlı modelleme üzerinden örnek problemler için çözüm süreçleri tasarlar.

Harekete Geç: Python ile etmen tabanlı bir model yapısı oluşturup uygular.

Yürüt: Eğitmenler öğrenciler ile etkileşimli bir biçimde etmen modelini çalıştırır ve genel çözüm akışını değerlendirir.

Karar Ver: Haftanın içeriğinin Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşimli kullanımı. Öğrenciler ayrıca etmen tabanlı uygulamalar ile elde edilen genel sonuçları inceler, tartışır.

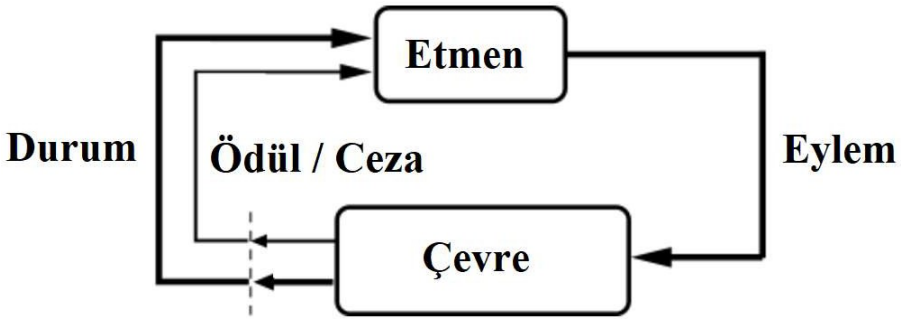
1. ALGILA

1.1. Etmen Tabanlı Yapay Zekâ Modelleme Temelleri

Eğitmene Not

Eğitmen, öğrencilere "**etmen**" ve "**etmen tabanlı yapay zekâ**" kavramları hakkında bilgi sahibi olup olmadıklarını sorar ve tartışır. Böylece öğrencilerin konuya dikkatini çeker.

Yapay Zekâ kapsamında çevresel faktörlerle etkileşim gerektiren problem çözümlemeleri için tasarlanmış çeşitli yaklaşımlar bulunmaktadır. Bu yaklaşımların özünde şu ana dek değinilen farklı Yapay Zekâ algoritmalarının etkileşimsel fonksiyonları içerecek şekilde tasarlanması yer almaktadır. Buna göre, çevreyle etkileşen Yapay Zekâ unsurları kısaca **etmen** ya da İngilizce karşılığından esinlenerek **ajan** (**agent**) olarak adlandırılmaktadır. Şekil 6.1'de tipik bir etmen modeli gösterilmiştir.



Şekil 6.1. Tipik bir etmenin genel yapısı (Web Kaynağı 6.1)

Şekil 6.1'den anlaşılacağı üzere bir etmen, çevresiyle etkileşim sonucunda elde ettiği bulgulara göre dönütlerde (yani eylemlerde) bulunabilecek şekilde tasarlanmaktadır. Bir etmenin eylemlerine karar verme aşamasında içerisinde bulunduğu durum ve çevreden aldığı ödül / ceza

dönütleri etkili olmakta, bunlar gelecek durumları da yönlendirmektedir. Buradan hareketle, özellikle kolektif, çok sayıda unsurun bir araya gelerek çözüm üretmesi gereken problem durumlarında birden fazla etmen de kullanılabilmektedir. Genel olarak bu yöndeki Yapay Zekâ tabanlı çözüm yaklaşımlarına **Etmen Tabanlı Yapay Zekâ**, bu yöndeki çözüm süreçlerinin tasarımına da **Etmen Tabanlı Modelleme** adı verilmektedir.

Eğitmene Not

Eğitmen öğrencilere, etmen tabanlı modellemenin aslında robotik sistemlere temel teşkil eden bir konu olduğunu ifade edecektir.

Benzer şekilde; bir insan olarak da çevreyle etkileşimlerimiz neticesinde kararlar alabilen ve eylemler gerçekleştiren unsurlar olarak etmen tabanlı modellemeyle olan benzerliklerimiz konusunda örnekler (örneğin markette kasiyer ile olan iletişimimiz, bir çocuğun sıcak bir unsura elini değdirdiği zaman ulaştığı tecrübe: 'sıcağa dokunmamalıyım'...vs.) vererek, etmen kavramına ilişkin pekiştirici örnekleri de vurgulayarak, öğrencilerin kavramları daha iyi anlamasını sağlayacaktır.

1.2. Etmen Tabanlı Modellemede Problem Tasarımı

Etmen tabanlı çözümlerin çalışacağı problemlerde, etkin çözüm elde etmek adına birtakım faktörlerin / bileşenlerin dikkate alınması ve bunlar altında tanımlamalar yapılması gerekmektedir.

Eğitmene Not

Eğitmen öğrencilere insan özellikleri ve eylemlerinden örnekler verir. Bu bağlamda şu yönde bir açıklama yapar: Örneğin, bir insanın göz rengi, boyu, kilosu gibi özellikler net değerler alan değişkenlerdir. Yürümek, koşmak, konuşmak gibi eylemler ise dinamik çözüm yolları; yani fonksiyonlar gibidir. Bu doğrultuda düşünerek, özellikler ve eylemleri birleştirmek suretiyle problemlere çözümler üretiriz.

Eğitmen ayrıca robot süpürge örneğini de irdeler. Robot süpürgeyi bir etmen olarak kabul edersek; kamera, kızılötesi gibi sensörler ile çevre algılama sağlanırken, dinamik tepki vericiler olarak çeşitli motorlar, fırçalar, mekanik kollar...vs. ile problem çözümü (çevreyi temizlemek / süpürmek) sağlanır.

İlgili faktörleri / bileşenleri genel olarak şöyle ifade edebiliriz:

- **Etmen Sayısı:** Çözümde kullanılacak etmenlerin sayısı, tek bir etmenin mi yoksa çok sayıda etmenin mi (kolektif bir şekilde) çalışacağı belirlenmelidir. Bazı problemler tek bir etmen ile kolayca çözülebilirken, bazı problemler de çok sayıda etmenin hem çevreyle

hem de birbirleriyle iletişim halinde olarak çalışmasını ve çözüme ulaşmasını gerekli kılmaktadır.

- **Etmen Nitelikleri:** Çözümde yer alacak etmenlerin sahip olacakları muhtemel parametreler aynı zamanda etmen nitelikleri olarak bilinmektedir. Bu parametreler, etmenlerin eylemlerine, muhakemelerine ve çevreyle ya da diğer unsurlarla (örneğin diğer etmenler) etkileşimlerine yön verecek, düzenlenebilir değerler olmaktadır.
- **Etmen Eylemleri:** Etmen eylemleri, ilgili etmenlerin çevreyle etkileşimleri ve geçerli nitelik değerleri üzerinden karar süreçlerini işletmelerini ve hatta eylemde bulunmalarını içermektedir. Bu eylem yapıları aslında birer fonksiyon olarak tanımlanmaktadır.
- **Çevre:** Etmenlerin içerisinde bulundukları çevrenin, en iyi etmen etkileşimleri için sınırları ve karakteristik nitelikleri ile tanımlanmaları gerekmektedir.

Eğitmene Not

Eğitmen öğrencilere 'etmen odaklı problemler başka neler olabilir?' şeklinde bir soru yönlendirir. Gelen cevaplara göre alternatif problem örneklerini öğrencilerle birlikte tartışır.

İfade edilen unsurlar ile, etmenlerin içerisinde bulunacağı problemin de tanımlanması gerekmektedir. Problemle ilgili olarak şu tanım faktörleri önemlidir:

- **Problem Kısıtları:** Problem ile ilgili çözümü temsil eden parametrelerin hangi kısıtlar altında olacağı, etmen davranışları neticesindeki çıktıların da gidişatını belirlemektedir.
- **Çevredeki Dinamik Unsurlar:** Etmenlerin eylemleri neticesinde durumları değişebilecek dinamik çevrenin söz konusu olacağı gibi, çevrede yer alacak ve etmenlerin gelecek davranışlarını şekillendirecek başka dinamik unsurlar da söz konusu olabilmektedir. Özellikle çok etmenli modellemelerde başka etmenler de dinamik unsurlar olarak kabul edilmektedir. Yine tek veya çok etmenli problem çözümlerinde çevreyle bağlantılı değişken parametreli unsurlar da problem çözümünü benzetim odaklı problemler için daha uyumlu hale getirebilmektedir.
- **Ödül / Ceza Fonksiyonları:** Etmenlerin eylemleri sonrasında gelecek kararlarını ve eylemlerini düzenleyecek birtakım çevresel ödüller veya cezalar dönüt olarak verilebilmektedir. Bu ödül ve cezalar çevredeki bazı unsurlarla etkileşimden doğabileceği gibi, etmenler için tanımlanan kurallarla ya da etmenin çevrede hareket halinde olduğu her aşamada uygulanabilmektedir.

Eğitmene Not

Eğitmen öğrencilere robot süpürgelerdeki problem tanım faktörlerinin neler olduğunu sorar ve şu doğrultuda tartışmayı yönlendirir: 'Evlerdeki robot süpürgeler hareket ettikleri sürece odaların haritasını oluşturabilmekte, aksi halde engellerle oda / ev sınırlarının inşa edilmesini sağlamaktadır. Sınırlar, dinamik unsurlar ve ödül / ceza fonksiyonları sayesinde problem kısıtları olarak algılanmakta ve robot süpürgenin davranışları neticesinde hem

temizlik / süpürme sağlanmakta, hem de harita oluşturularak gelecek eylemler için öğrenilmiş planlamalar yapılmaktadır.

Biliyor musunuz?

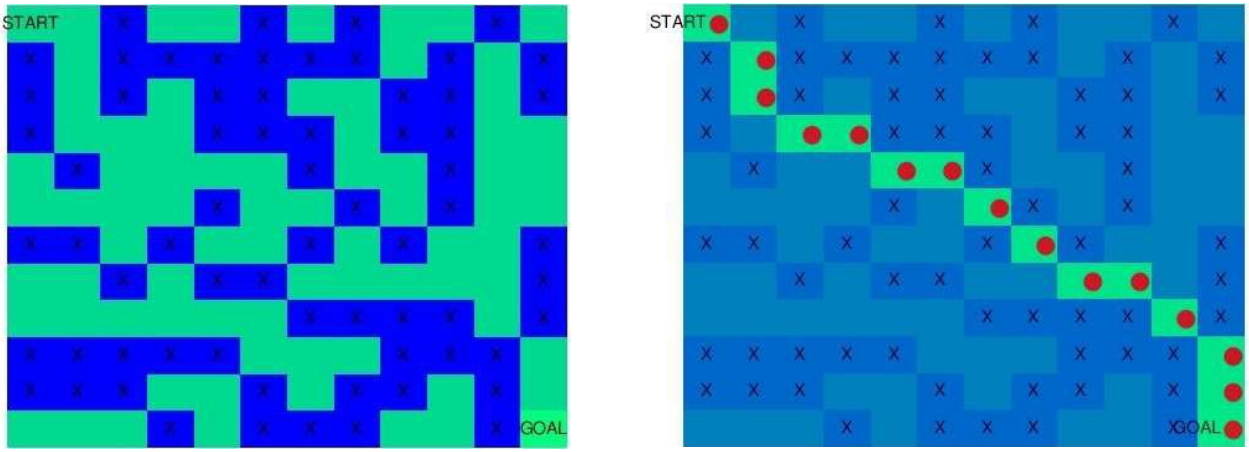
Etmen tabanlı modelleme anlaşılacağı üzere; insanların ve diğer canlıların dünya üzerinde çevreyle etkileşimine oldukça benzemektedir. Esasında bu benzetime dayanarak tasarlanan etmen tabanlı modelleme Yapay Zekâ'nın robotik uygulamalarında da oldukça önemli bir rol sahibidir. Yine Yapay Zekâ'daki bir başka alan olan zeki optimizasyon kapsamı da etmen tabanlı tasarımla ilişkilendirilebilmektedir. Buna göre, doğadaki canlı davranışlarının zeki optimizasyonda kullanılması etmen tabanlı tasarıma benzemektedir.

1.3. Q-Öğrenme ile Öğrenen Etmen Modelleme

Etmen tabanlı çözümlerde yapay zekânın makine öğrenmesi alanında kullanılan öğrenme yöntemi **takviyeli öğrenme (reinforcement learning)** olarak bilinmektedir. Takviyeli öğrenme gerçek hayattaki **yaşayarak öğrenme** gibidir. Bu öğrenmede genel olarak şu hususlar söz konusudur:

- Öğrenmede ödül / ceza mantığı vardır.
- Ödül olarak algılanacak değerler daha yüksekken, ceza olarak algılanan değerler düşük değerler olarak tasarlanır.
- Takviyeli öğrenme yapan unsurun ödül / ceza değerini belirleyen belli eylemler vardır (duvara çarparsan eksi 1 puan, çarpmazsan -yoluna devam edersen- her zaman artı 1 puan gibi).
- Takviyeli öğrenme yapacak unsur içerisinde bulunduğu ortam / problem için rastgele eylemlerde bulunur ve çok sayıda eylemlerle en iyi tecrübenin kazanılması sağlanır.

Etmen tabanlı çözümlerde takviyeli öğrenme için kullanılan temel tekniklerden biri de Q-Öğrenme (Q-Learning) olarak bilinmektedir. Q-Öğrenme'de daha önceden tasarlanmış ödül / ceza puanlarının yer aldığı bir tablo kullanılmak suretiyle, başlangıçta içi boş olan yeni bir eylem tablosunun en uygun değerlerle doldurulması sağlanır. Şekil 6.2a'da gösterildiği gibi; başlangıç noktası (start) ve bitiş noktası (goal) gösterilen bir problemde, engellerin olduğu her kareye negatif, engelsiz karelere ise pozitif puan verdiği düşünülürse; Q-Öğrenme ile gerçekleştirilen işlem sonucunda, etmenin Şekil 6.2b'de görüldüğü gibi kendisini amaca götüren en uygun yolu bulması sağlanmaktadır. **Bu örnekte soldaki görüntü önceden tasarlanmış ödül / ceza tablosuyken, sağdaki görüntü başlangıçta içi boş olan ama daha sonra en uygun yoldaki değerlerin diğer karelere göre daha yüksek olduğu eylem tablosudur.**



(a) (b)

Şekil 6.2. Q-Öğrenme ile örnek bir problem akışı

Q-Öğrenme'de etmen her seferinde rastgele adımları denemekte ve paragrafı takiben verilen formül sayesinde mevcut ödül / ceza tablosunu kullanmak suretiyle gelecek eylemlerinden gelebilecek maksimum değerleri (tıpkı satrançta gelecek hamleleri düşünmek gibi) dikkate almakta; yine öğrenmesini etkileyen diğer parametreleri ve yeni tablodaki mevcut ödül / ceza değerini harmanlayarak adım attığı eylemlerdeki değerlerin güncellenmesini sağlamaktadır. Böylelikle iyi eylemler zamanla daha fazla değer kazanırken, kötü eylemlerin değerlerinin zamanla azalması sağlanmaktadır.

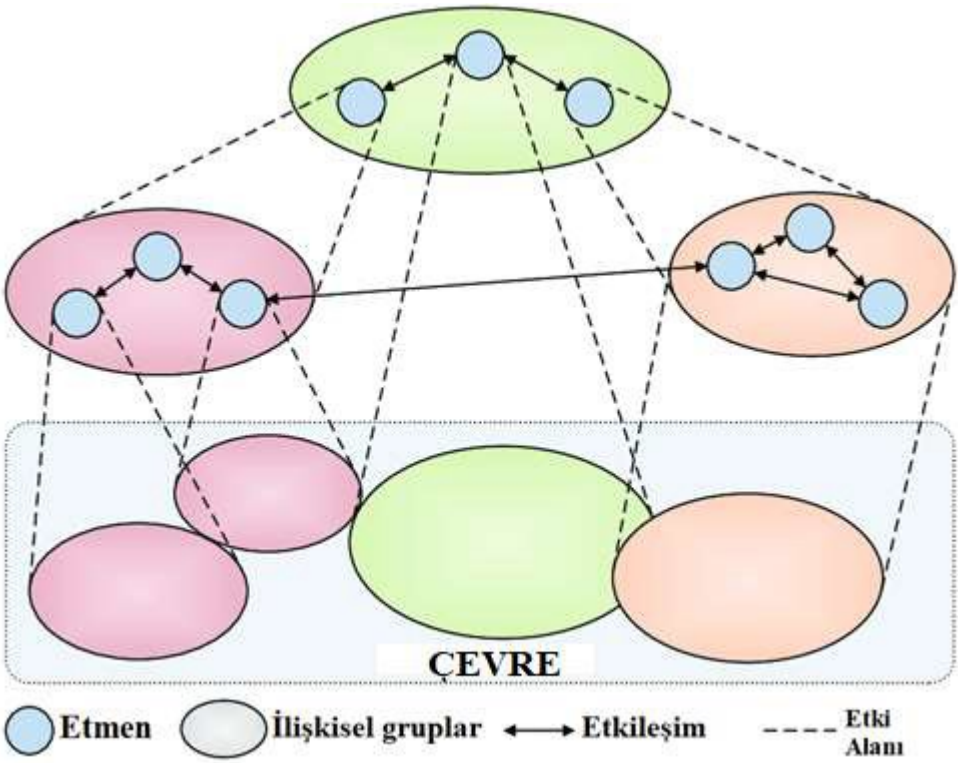
$$Q^{\text{yeni}}(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{eski değer}} + \underbrace{\alpha}_{\text{öğrenme oranı}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{ödül}} + \underbrace{\gamma}_{\text{azalma değeri}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{gelecekteki tahmini maks. değer}} \right)}^{\text{eylem sonucunda öğrenilen değer}}$$

Eğitmene Not

Eğitmen öğrencileri Q-Öğrenme formülünün detaylarını öğrenmeye zorlamadan, formülün parantezlerle gösterilen kısımlarını yaşayarak öğrenmeyi düşünerek ('bu eylemi yapmak her zaman için iyidir, ben de şimdiye kadar bu eylemi yaparak olumlu dönütler aldım, buna dayanarakgelecekteki hangi eylemlerle daha fazla olumlu dönüt alırım?') mantıksal çerçevede anlatır.

1.4. Çok Etmenli Etkileşimler

Birden fazla etmenin varlığı çok etmenli modellemeleri ortaya çıkarmaktadır. Bu modelleme özellikle günümüz karmaşık problemlerinin çözümünde daha etkin sonuçlar üretebilmektedir. Çok etmenli problem modellemelerinde etmenler arası ilişkiler, her bir etmenin çevreyle etkileşimi ve bağlı oldukları çevresel düzenlemeler farklılıklar içerebilmekte, böylece karmaşık düzendeki problemlere adaptasyon daha kolay sağlanabilmektedir.



Şekil 6.3. Çok etmenli problem ortamı (Web Kaynağı 6.2)

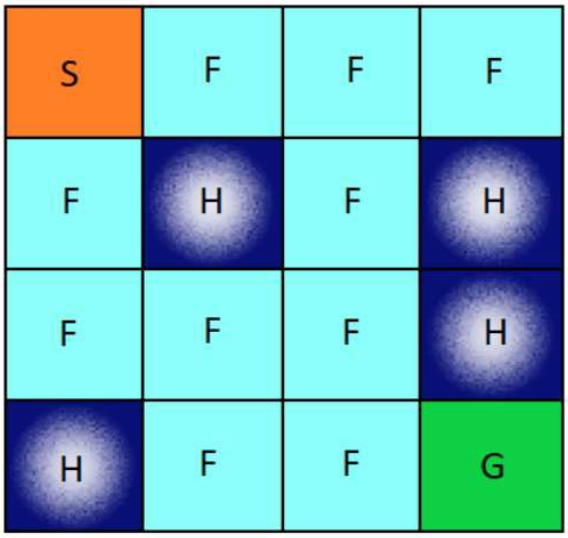
Eğitmene Not

Eğitmen, çok etmenli modellemeye örnek olarak farklı karakterlerin birbiriyle etkileşimde olduğu video oyunlarını örnek verir. Bu noktada, Braw stars ya da platform tabanlı video oyunlarda (örneğin, Mario), karakterlerin nitelikleri, eylemleri, çevrenin düzenlenmesi ve hatta çok etmenli mantıkta farklı karakterler arası ilişkileri etmen tabanlı modelleme odağında öğrenciler ile tartışır.

2. TASARLA

Eğitmene Not

Eğitmen, tek etmenli modelleme kapsamında çözümlenebilecek **donmuş göl (frozen lake)** adlı örnek problemi öğrencilere açıklar. Buna göre Şekil 6.4'te gösterildiği gibi, sol üst köşeden (**S**) başlayıp sağ alttaki bitiş noktasına (**G**) ulaşmak için donmuş yollardan (**F**) geçmek bu sırada ve çukurlara (**H**) düşmemek gerekmektedir.



Şekil 6.4. Donmuş göl probleminin ortam modellemesi

Q-Öğrenme odaklı bakıldığında etmen eylemlerinin donmuş (**F**) kutucukları için normal değerde, çukur kutucukları (**H**) için düşük değerde ve ulaşılması gereken bitiş noktası (**G**) için yüksek değerde olması gerekmektedir.

Eğitmene Not

Eğitmen, uygulamaya geçmeden önce Python kodlama için yaygın bir biçimde kullanılan OpenAI Gym kütüphanesi hakkında öğrencilere bilgi verir. Söz konusu kütüphane özellikle farklı etkileşimsel problemleri ve etmen tabanlı modelleri içerisinde hazır tutan kod yapıları sayesinde geliştiricilere özellikle takviyeli öğrenmeye dayalı etmenler tasarlamalarına imkân vermektedir.

Problemin çözümü için problem puan tablosu ve ortamının hazır olduğu **OpenAI** etmen tabanlı modelleme kütüphanesi olan **gym** kod ortamı için Python'a yüklenir:

pip install gym

3. HAREKETE GEÇ

Etmen tabanlı modelleme kütüphanesi **gym**'in yüklenmesi sonrasında, kod ortamında kullanılacak gym kütüphanesi çağrılır ve Donmuş Göl problemi tanımlamasıyla birlikte boş Q tablosu oluşturulur. Yine Q-Öğrenme parametrelerinin belirlenmesi ile süreçte ihtiyaç duyulan ödül değerleri listesinin değişken tanımları da kodlanır:

```
1 import gym
2 import numpy as np
3 ortam = gym.make('FrozenLake-v1', is_slippery=False)
4 Q = np.zeros([ortam.observation_space.n,ortam.action_space.n])
5 eta = .628
6 gma = .9
7 dongu = 5000
8 odul_list = []
```

Şekil 6.5. Etmen modelleme kütüphanesi ve problemin yüklenmesi

1. ve 2. satırlar gerekli olan **gym** ve **numpy** çağrılarına tekabül etmektedir.

3. satırda Donmuş Göl problemi ile **ortam** adı altında problem ortamının otomatik modellenmesi sağlanır.

4. satırda içeriği boş Q tablosu tanımlanır. Böylelikle, Şekil 6.4'te gösterilen 4x4'lük hücre yapısı dikkate alınmak suretiyle her satırda soldan sağa doğru birer hücrenin, her sütunda da sırasıyla **sola, aşağıya, sağa** ve **yukarıya** hareketlerin puan değerlerini tutacak toplamda dörder sütunun olduğu, toplamda 16x4'lük bir Q tablosu oluşmuş olur.

5.-8. satırlar arasında Q-Öğrenme parametrelerini, toplam öğrenme döngü sayısını ve öğrenme sırasında puan değerlerini tutacak ilgili değişkenlerin tanımlanması sağlanır.

Devam eden kod düzeninde Q-Öğrenme tekniğinin kodlanması sağlanır:

```
 9 for i in range(dongu):
10     s = ortam.reset()
11     odul_tumu = 0
12     d = False
13     j = 0
14     while j < 99:
15         ortam.render()
16         j+=1
17         a = np.argmax(Q[s,:] + np.random.randn(1,ortam.action_space.n)*(1./(i+1)))
18         s1,o,d,_ = ortam.step(a)
19         Q[s,a] = Q[s,a] + eta*(o + gma*np.max(Q[s1,:]) - Q[s,a])
20         odul_tumu += o
21         s = s1
22         if d == True:
23             break
24     odul_list.append(odul_tumu)
25     ortam.render()
```

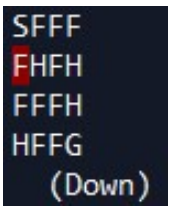
Şekil 6.6. Q-Öğrenme algoritmasının kodlanması

9. satır itibariyle toplam öğrenme döngü sayısı kadar devam edecek Q-Öğrenme tekniği adımları kodlanır.

10.-13. satır arasında öğrenme süreci öncesinde ilgili değerlerin ve problem ortamının başlangıç konumlarına alınması sağlanır.

14. satır ve sonrasında etmenin her adımında Q hesaplamalarının yapılması (19. satır) ve nihayetinde Q tablosunun adım adım güncellenmesi sağlanır. Ayrıca **odul_tumu** değişkenine hareketler sonunda elde edilen puan da eklenmektedir.

25. satırda ilgili döngü sonunda etmenin ortamdaki hareketi Şekil 6.7'de gösterildiği gibi ekrana yansıtılmaktadır (Ortam görüntüsünde etmen, kırmızıyla gösterilen hücre noktasındadır. Yine etmenin bir sonraki hareketi ortam görüntüsü altında ifade edilmektedir. Buna göre şekilde gösterildiği an itibariyle, ikinci satır ilk sütunda donmuş bir yol üzerinde yer alan etmen bir sonraki hareketini aşağı hareket etme yönünde tercih etmiştir).



Şekil 6.7. Donmuş Göl probleminde etmen hareketleri

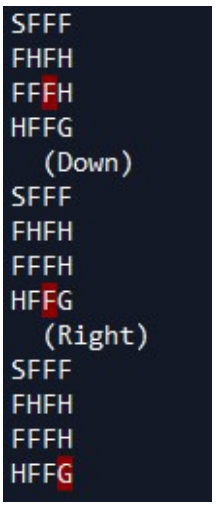
4. YÜRÜT

Q-Öğrenme tekniği kodlarının çalışması sonrasında ekrana nihai Q tablosunu (dolayısıyla Donmuş Göl ortamında hareket durumunu) ve ortalama puan değerini yazdıran kodlar yazılır:

```
26 print("Elde edilen Q Tablosu:")
27 print(Q)
28 print("Ortalama ödül değeri:" + str(sum(odul_list)/dongu))
```

Şekil 6.8. Q tablosu ve ortalama ödül değerinin görüntülenmesi

Bütün kod düzeni çalıştırıldığında kod içeriğinde belirtilen 5000 adımlık döngüde, etmenin bitiş noktasına ulaşmak adına gerçekleştirdiği adımlargözlemlenmiş olur:



Şekil 6.9. Q-Öğrenme sürecinde etmen adımlarının problem üzerine yansıması

Eğitmene Not

Eğitmen öğrencilere her hareket neticesinde etmenin amaca ulaşıp ulaşmadığını, hareketlerin ne yönde puanlar (pozitif / negatif) getirebileceğini sorarak; nihai Q tablosu öncesinde tartışır.

5. KARAR VER

5.1. Sonuçların Gözlemlenmesi

Q-Öğrenme süreci tamamlandığında ekranda Q tablosu ve ortalama puan değeri gözlemlenir. (İlgili kod bütünü, Github platformu Hafta6 klasörü altında, H6_etmen_frozenlake.py dosyası ile sunulmuştur):

```
Elde edilen Q Tablosu:
[[0.      0.      0.59049 0.     ]
 [0.      0.      0.6561  0.     ]
 [0.      0.729   0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.81    0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.9     0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      1.      0.     ]
 [0.      0.      0.      0.     ]]
Ortalama ödül değeri:0.9936
```

Şekil 6.10. Nihai Q tablosu ve ortalama ödül değeri

Eğitmene Not

Elde edilen Q tablosu görüntüsünde (Şekil 6.10) her satırda soldan sağa doğru birer durumun ve her sütunda da sırasıyla **sola, aşağıya, sağa** ve **yukarıya** hareketlerin puan değerleri gösterilmektedir. Örneğin birinci satır başlangıç hücresi (**S**) içindir. Bu satırda üçüncü sütunda en yüksek puan değeri yer aldığı için etmen ilgili hücreden **sağa doğru** hareket edecektir. **Sağa doğru hareket edildiğinde ulaşılan hücre indeksi 1 olması nedeniyle tabloda bir alt satıra bakılabilir. Yine en yüksek sütun değerlerine göre gidilen yeni hücrenin soldan sağa doğru sayılmasıyla elde edilen indeks numarasına tekabül eden satır gözlemlenerek** öğrenciler ile başlangıç noktasından bitişe doğru etmenin hangi hareketi gerçekleştirmeyi öğrendiği anlaşılır.

PYTHON KODLARI:

```
import gym
import numpy as np
ortam = gym.make('FrozenLake-v1', is_slippery=False)
Q = np.zeros([ortam.observation_space.n,ortam.action_space.n])
eta = .628
gma = .9
dongu = 5000
odul_list = []
for i in range(dongu):
    s = ortam.reset()
    odul_tumu = 0
    d = False
    j = 0
    while j < 99:
        ortam.render()
        j+=1
        a = np.argmax(Q[s,:] + np.random.randn(1,ortam.action_space.n)*(1./(i+1)))
        s1,o,d,_ = ortam.step(a)
        Q[s,a] = Q[s,a] + eta*(o + gma*np.max(Q[s1,:]) - Q[s,a])
        odul_tumu += o
        s = s1
        if d == True:
            break
    odul_list.append(odul_tumu)
    ortam.render()

 print("Elde edilen Q Tprint(Q)
print("Ortalama ödül değeri:" + str(sum(odul_list)/dongu))
```

Dünya'dan Haberler

İnsansı robotlar bilimi sevdirecek

Cumhurbaşkanlığı Millet Kütüphanesi Teknoloji Atölyesi'nde 13-15 yaş grubundaki çocukları bilim ve teknoloji çalışmalarına yönlendirmek amacıyla 'insansı robot' eğitimi verilmeye başlandı.

Atölyelerde altı kişilik yapılan etkinlikte, insansı gövde ve eklem yapısına sahip robotların çalışma prensipleri ile işlevleri ele alınıyor. Cumhurbaşkanlığı Millet Kütüphanesi Eğitim Merkezi ve Atölyeler Koordinatörü Ayhan Bozkurt, TÜBİTAK tarafından kurulan bilim ve teknoloji atölyelerinde çocuklara, robotların ne olduğunu anlatacaklarını belirtti. Bozkurt, şöyle devam etti:

"*İki tane insansı robot üzerinden gözlem yapacaklar. Bizim sorularımız olacak. Onlara, 'biz robot olsaydık bizi nasıl programlarlardı, hangi direktifleri vereceklerdi' gibi sorularla insansı robotlara yönelik farkındalık oluşturmak istiyoruz. Amacımız, çocukların bilim ve teknolojiye olan meraklarını artırmak, onların o alanda çalışmalar yapmalarını sağlamak ve teşvik etmektir. Onların gerçek anlamda bir robot görerek bu tarz çalışmalarla bilime ve teknolojiye yönelmelerini sağlamaktır.*" Daha önceki eğitimlerde, çocukların robotlara yönelik ilgilerinin fazla olduğunu gördüklerini aktaran Bozkurt, etkinliklerin devam edeceğini bildirdi (Web Kaynağı 6.3).

6. İLAVE ETKİNLİK

Eğitmene Not

Eğitmen, öğrencilere ceza / ödül fonksiyonuna sahip olmayan; basit yapıda çok etmenli bir model tasarımı geliştireceklerini belirtir.

Bu örnekte etmenlerin aralarında sayısal değer (varlık) paylaşımı yaptığı bir akış kurgulanmıştır.

Öğrenciler, ilk olarak **pip install mesa** komutu ile etmen tabanlı modelleme için kullanılan, mesa adlı bir başka Python kütüphanesini kurarlar (Web Kaynağı 6.4).

Ardından paragrafı takiben verilen komutları kodlayarak standart bir etmen modelini tasarlarlar (Söz konusu uygulama, Github platformu Hafta6 klasörü altındaki iki dosya altında: H6_etmen_2_model-kodu.py ve H6_etmen_2_model-ornegi.py ile paylaşılmış durumdadır):

Şekil 6.11'de gösterildiği gibi öğrenciler gerekli kodları yazar.

1 nolu kod satırında yapay zekâ kütüphanesi olan **mesa** kütüphanesinin **Agent ve Model** sınıflarını çağırır.

2 nolu kod satırında **"Etmen"** isminde bir sınıf oluşturur.

4-5-6 nolu kod satırında **"Etmen"** sınıfının içerisine **def__init__** isminde bir fonksiyon ile her bir etmen için id tanımlar.

7 nolu kod satırında **"Cevre"** isminde bir sınıf oluşturur.

9-10 nolu kod satırında **"Cevre"** sınıfının içerisine **def__init__** isminde bir fonksiyon ile N adet etmen sayısı tanımlanır.

12-13 nolu kod satırında oluşturulan bir for döngüsü ile etmen sayısı kadar döngü Etmen sınıfı çağrılarak a değişkenine aktarılır.

```
1 from mesa import Agent, Model
2 class Etmen(Agent):
3     """Başlangıç varlık değerine sahip bir etmen."""
4     def __init__(self, unique_id, model):
5         super().__init__(unique_id, model)
6         self.varlik = 1
7 class Cevre(Model):
8     """Etmenlerin yer alacağı çevre."""
9     def __init__(self, N):
10        self.num_agents = N
11        # etmenlerin oluşturulması
12        for i in range(self.num_agents):
13            a = Etmen(i, self)
```

Şekil 6.11. Başlangıç varlığı ve etmenlerin yer alacağı çevre kod ekranı

Eğitmene Not

Eğitmen öğrencilerden, bu notu takip eden şekildeki yeni kodları da eklemelerini ister. Her kod satırında kodun ne anlama geldiğini sorar ve karşılıklı tartışır.

```python
from mesa import Agent, Model
from mesa.time import RandomActivation
class Etmen(Agent):
    """Başlangıç varlık değerine sahip bir etmen."""
    def __init__(self, unique_id, model):
        super().__init__(unique_id, model)
        self.varlik = 1
    def step(self):
        # Etmenin her adımda gerçekleştireceği eylem
        print ("Merhaba, ben etmen " + str(self.unique_id) +".")

class Cevre(Model):
    """Etmenlerin yer alacağı çevre."""
    def __init__(self, N):
        self.num_agents = N
        # etmenlerin oluşturulması
        for i in range(self.num_agents):
            a = Etmen(i, self)
            self.schedule.add(a)
    def step(self):
        '''Etmene bir adım attır.'''
        self.schedule.step()
```

Şekil 6.12. Etmen ve çevre modeli kodlanması

Yeni eklenen kodlarda **RandomActivation** kütüphane bileşeni ile etmenlerin çevre içerisinde bir adımlık eylem gerçekleştirmeleri sağlanabilmektedir. Eylemlerde her adım **step** fonksiyonu ile karşılanmakta, her adımda herhangi bir etmen kendi sıra numarasını söylemektedir.

Eğitmene Not

Eğitmen, sonraki aşamada öğrencilere yeni bir Python kod dosyası açtırarak, önceki aşamada kodlanan etmen modelinin kaydedilip, yeni kod ortamına **import** edilmesini sağlar.

```python
from etmen_modeli import Etmen
yeni_model = Etmen(10)
yeni_model.step()
```

Şekil 6.13. Adımlar eşliğinde etmen modellenmesi

Bu kod örneği ile rastgele 10 etmenin otomatik olarak oluşturulması sağlanır.

Çalıştırılan kod örneği ile kendilerini ifade eden rastgele 10 etmenin oluşturulduğu gözlenir (Şekil 6.14):

```
Merhaba, ben etmen 2.
Merhaba, ben etmen 9.
Merhaba, ben etmen 5.
Merhaba, ben etmen 3.
Merhaba, ben etmen 7.
Merhaba, ben etmen 0.
Merhaba, ben etmen 4.
Merhaba, ben etmen 6.
Merhaba, ben etmen 8.
Merhaba, ben etmen 1.

In [9]:
```

Şekil 6.14. Çalıştırılan kod örneği sonucunda çalışan farklı etmenlerin çalışması

Sonraki aşamada, kendi değerini (varlık) kontrol eden ve eğer varsa rastgele başka bir etmene bağış yapan bir fonksiyon yapısı, önceden yazılan ve etmen model dosyası içerisinde yer alan **step** fonksiyonu düzenlenerek eklenir:

```
20    def step(self):
21        if self.varlik == 0:
22            return
23        baska_etmen = self.random.choice(self.model.schedule.agents)
24        baska_etmen. varlik += 1
25        self. varlik -= 1
```

Şekil 6.15. Etmenler arası etkileşimin kodlanması

Söz konusu ekleme sonrası modelin import edildiği yeni kod dosyası içerisinde 10 adet etmen oluşturma koduna ek olarak etmenlerin 10 adımlık eylemde bulunmasını sağlayan kod yapısı eklenir:

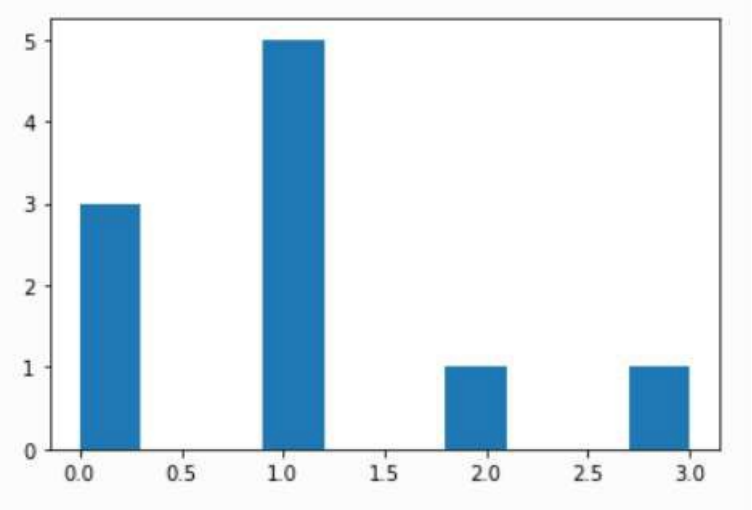
```
1 from etmen_modeli import Etmen
2 yeni_model = Etmen(10)
3 for i in range(10):
4     yeni_model.step()
```

Şekil 6.16. Bütün etmenler için döngüsel akışın kodlanması

Öğrenciler, yeni **step** fonksiyonu neticesinde çalıştırılan kod yapısı ile her bir etmenin 10 adım sonunda sahip oldukları değerlerin grafiksel gösterimini görüntüler. Bunun için **matplotlib.pyplot** kütüphanesi ile birlikte etmenlerin son değerlerinin bir araya toplandığı değişken üzerinden görüntüleme yapılır.

```
 9 etmen_varlik = [a.varlik for a in yeni_model.schedule.agents]
10 plt.hist(etmen_varlik)
```

Şekil 6.17. Etmen hareketlerinin görselleştirilmesi



Şekil 6.18. 10 adımlık etkileşim sonrasında değerlerin dağılımı

PYTHON KODLARI:

```
#model kod yapısı
from mesa import Agent, Model
from mesa.time import RandomActivation
class Etmen(Agent):
    """Başlangıç varlık değerine sahip bir etmen."""
    def _init_(self, unique_id, model):
        super()._init_(unique_id, model)
        self.varlik = 1
    def step(self):
        # Etmenin her adımda gerçekleştireceği eylem
        print ("Merhaba, ben etmen " + str(self.unique_id) +".")
class Cevre(Model):
    """Etmenlerin yer alacağı çevre."""
    def _init_(self, N):
        self.num_agents = N
```

```
        # etmenlerin oluşturulması
        for i in range(self.num_agents):
            a = Etmen(i, self)
            self.schedule.add(a)
    def step(self):
        if self.varlik == 0:
            return
        baska_etmen = self.random.choice(self.model.schedule.agents)
        baska_etmen. varlik += 1
        self. varlik -= 1
```

```
#örnek problem çözümü
from etmen_modeli import Etmen
yeni_model = Etmen(10)
for i in range(10):
    yeni_model.step()
import matplotlib.pyplot as plt
etmen_varlik = [a.varlik for a in yeni_model.schedule.agents]
plt.hist(agent_varlik)
```

Eğitmene Not

Eğitmen, ilgili hafta eğitimine dair öğrencilerin ilgisini arttırmak ve yeni öğrenilen bilgileri önceden öğrenilen bilgiler ile karşılaştırmak için aşağıda maddeler halinde verilen soruları tartışır.

- İlgili haftada ele alınan yapay zekâ uygulamalarına uygun örnek veri setleri neler olabilir?
- İlgili haftada ele alınan veri setlerinin yapay zekâ teknik ve kütüphanelerinde ne derecede başarım etkisi olmuştur?
- İlgili haftada öğretilen yapay zekâ tekniği ile uyumlu ve uyumsuz olabilecek açık erişimli veri seti çeşitleri sunan (kaagle, github vs.) internet sitelerinden elde edilecek farklı veri çeşitleri ile yapay zekâ tekniği uyumluluğu nasıldır?

.

Eğitmene Not

Eğitmen, öğrencilerin 5. ve 6. Hafta içerisinde elde ettikleri bilgi-becerileri değerlendirmek, aktif katılımlarını sağlamak ve onları gelecek haftalara motive etmek için 'Kahoot' uygulamasını ya da benzeri bir Web 2.0 uygulamasını kullanarak 'bilgi yarışması' düzenleyebilir.

5. ve 6. Hafta bağlamında sorulabilecek sorular; Yapay Sinir Ağları temelleri / uygulamaları, Etmen Tabanlı Modelleme yaklaşımı ve ve Etmen Tabanlı Modelleme ile uygulamalar yönünde olabilecektir.

Kaynakça

Web Kaynağı 6.1: http://www.incompleteideas.net/book/ebook/node28.html

Web Kaynağı 6.2: https://www.researchgate.net/publication/324808821_Self-Reconfigurable_Manufacturing_Control_based_on_Ontology-Driven_Automation_Agents

Web Kaynağı 6.3: https://www.hurriyet.com.tr/yerel-haberler/ankara/insansi-robotlar-bilimi-sevdirecek-41850836

Web Kaynağı 6.4: https://mesa.readthedocs.io/en/master/tutorials/intro_tutorial.html

7. Hafta: Zeki Optimizasyon

Ön Bilgi:

- Optimizasyon kavramının temelleri.
- Python kodlamada döngüler, fonksiyonlar ve kütüphane yapılarının kullanımı (Bu konuda Python anlatım videoları: 10-11. Döngüler, 13. Fonksiyonlar, 14. Python Kütüphane Kullanımı yardımcı araç olarak kullanılabilecektir).

Haftanın Kazanımları:

- Öğrenciler optimizasyon kavramını kavrar.
- Öğrenciler optimizasyon tabanlı problem modelleri tasarlayarak uygular.
- Öğrenciler optimizasyona dayanan çözüm süreçlerini yorumlamayı kavrar.
- Öğrenciler yapay zekâ alanı kapsamındaki zeki optimizasyon yöntemini anlar.
- Öğrenciler Python programlama dilini kullanarak genetik algoritma program kodunu yazabilir.
- Öğrenciler Python programlama dilini kullanarak karınca koloni optimizasyonu program kodunu yazabilir.
- Öğrenciler, örnek bir probleme göre zeki optimizasyon çözümü gerçekleştirme yeteneği kazanır.
- Öğrenciler zeki optimizasyon algoritmalarının davranışlarını değerlendirerek yorumlar.
- Öğrenciler zeki optimizasyon süreçlerinin Yapay Zekâ alanı açısından önemini kavrar.

Haftanın Amacı:

Bu haftanın amacı, “zeki optimizasyon” kavramının tüm öğrenciler tarafından doğru bir şekilde kavranmasını sağlamaktır. Haftanın bir diğer amacı da zeki optimizasyona temel teşkil eden doğa esinli kolektif davranışlarla zeki optimizasyon tasarlama mantığını yorumlama noktasında istendik bilgi birikimini aktarmaktır. Bu kapsam içerisinde öğrencilerin Python programlama dilini kullanarak Genetik Algoritma ve Karınca Koloni Optimizasyonu olarak bilinen önemli zeki optimizasyon algoritmalarının kullanımı konusunda yeterlikler elde etmesi de sağlanacaktır.

Kullanılacak Malzemeler:

Bilgisayar, kâğıt, kalem, Python kod editörü, örnekler için çıktı görselleri

Haftanın İşlenişi:

Algıla: Optimizasyon ve zeki optimizasyon kavramlarını öğrenir.

Tasarla: Farklı zeki optimizasyon algoritmalarıyla örnek problemler için çözüm süreçleri tasarlar. Bu hafta için Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşim Tasarla aşaması ile başlatılabilmektedir.

Harekete Geç: Python ile zeki optimizasyon tabanlı çözüm yaklaşımı oluşturup uygular.

Yürüt: Eğitmenler öğrenciler ile etkileşimli bir biçimde farklı zeki optimizasyon algoritmalarını çalıştırır ve genel çözüm akışını değerlendirir.

Karar Ver: Haftanın içeriğinin Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşimli kullanımı. Öğrenciler zeki optimizasyon süreçleriyle elde ettikleri sonuçları yorumlar.

1. ALGILA

1.1. Optimizasyon Kavramı

Eğitmene Not

Eğitmen, öğrencilere "**optimizasyon**" kavramının ne anlama geldiğini sorar ve tartışır. Gerçek hayattan örneklerle bilinen birtakım çözümlerin aslında optimizasyonla ilişkili olduğunu açıklar. Böylece öğrencilerin konuya dikkatini çeker.

Optimizasyon ya da başka bir deyişle 'en iyileme' kavramı, genel olarak bir problemin çözümünde en uygun parametrelerin tespit edilmesi olarak bilinmektedir. Optimizasyon aslında günlük hayatta sayısal tahminlerde bulunurken ya da aklımızdan planlamalar yaparken gerçekleştirdiğimiz çözümlerin bilimsel ifade ediliş şeklidir. Daha genel anlamda optimizasyon kavramının matematik ile bağlantısı, değeri bilinmeyen denklem değişkenlerinin bulunması şeklinde olmaktadır.

Matematiksel anlamda optimizasyon, denklemlerle modellenmesi yapılmış bir problem için bilinmeyen değerlerin bulunmasına karşılık gelmektedir.

Optimizasyonu matematiksel olarak ifade etmek için fonksiyon gösterimleri kullanılmaktadır. Örneğin, y ile gösterilen yazılı sınavı puanı ve s ile gösterilen sözlü sınavı puanı dikkate alınmak üzere, y'nin %60'ı ve s'nin %40'ının alınmasıyla hesaplanacak ders başarı puanı DB şu şekilde gösterilebilmektedir:

DB(y, s) = (0,6 * y) + (0,4 * s)

DB fonksiyonunda bir dersten başarılı olmak için başarı puanının en az 60 olması gerektiğini kabul edersek, yazılıdan (y) 50 alan bir öğrencinin, sözlüden (s) en az kaç alması gerektiğini bulmak, optimizasyona tekabül etmektedir. Burada denklemdeki 0,6 ve 0,4 **katsayı** olarak bilinmekte, y ve s ise **değişken** olarak ifade edilmektedir. Eğer fonksiyona +10 puan kanaat puanı eklenecek olursa bu değer de **sabit** olarak isimlendirilmektedir.

Eğitmene Not

Eğitmen öğrencilere, 50 yazılı puanı karşısında, sözlüden en az 75 puan alınırsa başarılı olunacağını, 75'den daha fazla puanların da bu kapsamda kabul edilebileceğini açıklar. Bununla birlikte öğrencilerden benzeri bir matematiksel model tasarlayıp tasarlayamayacaklarını sorar ve tartışır.

DB fonksiyonunun hesaplanması aklımızdan çözebileceğimiz bir optimizasyon problemidir. Bununla birlikte gerçek dünyadaki birçok problem optimizasyon modellenmesiyle çözümlenebilmektedir. Örnek olarak:

- Eldeki hammaddeleri en az kayıpla kullanıp istenilen ölçülerde ürün elde etmek bir optimizasyon problemidir.
- Bir inşaatta istenilen yükseklik ve genişlikteki odalarda, en iyi güneş ışığı oranını elde etmek için belli sayıdaki pencerelerin hangi noktalara konumlanması gerektiği optimizasyon problemi olmaktadır.
- Belli özellikleri baskın kimyasal bir maddeyi elde etmek için yüzlerce bileşenden kaçar gram karıştırılması gerektiği optimizasyon çözümlemesini gerekli kılmaktadır.
- 10 farklı şehirden hangilerinin ziyaret edilmesi neticesinde en kısa mesafe ile en az yakıt tüketiminin sağlanacağı sorusu bir optimizasyon çözümünü işaret etmektedir.

Özellikle mühendislik tabanlı optimizasyon problemleri daha yüksek dereceli olabilmekte (yani değişkenler daha yüksek üstel sayılarda temsil edilmekte), aynı anda çok daha fazla değerin bulunmasını gerekli kılabilmekte ve daha karmaşık fonksiyonlara tekabül edebilmektedir.

Optimizasyon problemleri modellenirken şu hususlar da oldukça önemli olmaktadır:

- Değeri bulunması gerekli olan değişkenlerin sınır değerleri dikkate alınmalıdır. [Örneğin, DB probleminde yazılı (y) ve sözlü sınav (s) değişkenleri en az 0 (sıfır), en fazla 100 (yüz) olmalıdır.]
- Aynı anda birbirleriyle ilişkili birden fazla fonksiyonun dikkate alınması gerekebilmektedir. Örneğin bir şirkette K fonksiyonu ile karı yükselten değişkenler bulmaya çalışılırken, aynı anda Z fonksiyonu ile zararı en az tutmaya çalışan değişkenler bulunabilmekte; aynı anda iki fonksiyonda da yer alan personel sayısı şeklindeki bir değişken için her iki fonksiyonu da sağlayan en uygun (optimum) değer bulunabilmektedir.
- Bazı optimizasyon problemleri doğrudan bilinmeyen değerlerin bulunmasına odaklanırken (sürekli optimizasyon) bazıları bilinen çözümlerin hangi kombinasyonlarının daha iyi olacağını bulmak için (kombinasyonel optimizasyon) modellenmektedir.

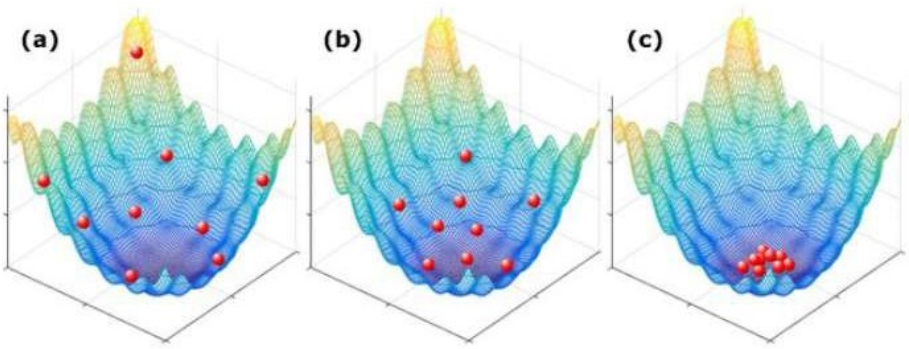
1.2. Zeki Optimizasyon Kavramı

Matematikte bilinen klasik optimizasyon yöntemleri, karmaşıklığı yüksek ve oldukça fazla değişken içeren problem modellerinde başarılı sonuç verememektedir. Bunun neticesinde temellerini doğadaki canlı sürülerinin iş birliği içerisindeki, şans faktörü içeren eylemlerinden alan yapay zekâ tabanlı optimizasyon algoritmaları geliştirilmiştir.

Zeki optimizasyon, makine öğrenmesi kapsamında olduğu gibi öğrenme süreci gerektirmemektedir. Daha çok döngüler yardımıyla en uygun değerlerin tespit edilmeye çalışılmasına dayanmaktadır. Bunu sağlamak için de algoritmalar tasarlanırken şu mekanizmalar işletilmektedir:

- Tıpkı etmen tabanlı modellemede olduğu gibi *N* sayıda etmenin tanımı yapılmaktadır. Bu noktada etmen tabanlı modellemeden farklı olarak bu çözüm unsurları sadece sayı ya da kombinasyon aramakla sorumludur. Her etmen algoritma başlangıcında rastgele değerlerle eşlenerek aramaya başlamaktadır. **Bu süreç tıpkı çok sayıda kişinin aynı problemi çözmek için uğraşıp elde ettikleri sonuçlara göre birbirleriyle yardımlaşmaları gibidir.**
- Problemle bağlantılı olarak *N* sayıda etmen rastgele hareketlerle problem modeline (denkleme / fonksiyona) konulacak değerler türetmekte ve her bir etmenin hesapladığı nihai değerlerle en uygun (en küçük / minimum ya da en büyük / maksimum) sonucu veren değerler tespit edilmektedir.
- *N* adet etmen arasında belli bir aşamada en uygun değeri verene diğerleri sayısal olarak yaklaştırılmaya çalışılmakta, bu süreç belli bir döngü sayısı boyunca işletilmektedir.
- Zeki optimizasyonu sağlamak için farklı matematiksel mekanizmalarla tasarlanan ve doğadaki canlılardan (kuşlar, böcekler…vs.), insan topluluklarından ve hatta dinamik olgulardan (mevsimsel değişimler, fizik kanunlarıyla bağlantılı olaylar, çiçek tozlaşması, akarsuların akışı…vs.) esinlenen, bilinmeyen değerleri arama yolunda kendi iç farklılıklarını içeren algoritmalar tasarlanabilmektedir. **Farklı algoritmalarda *N* adet çözüm unsurlarını tarif etmek için algoritma esin kaynağına göre parçacık, birey, arı, karınca gibi isimler kullanılabilmektedir.**

Optimizasyon içerisinde tasarlanan modeller görsel olarak incelendiğinde Şekil 7.1'dekine benzer bir çözüm süreci elde edilmiş olmaktadır. Şekilde görüldüğü gibi, bulunmaya çalışılan en uygun düşük değeri bulmak soldan sağa doğru belli bir süreci gerekli kılmaktadır. Bu sırada istenilen değere ulaşma sırasında çeşitli engeller ve yanlış tespitlerde bulunmak da olasıdır. Bu nedenle zeki optimizasyon garanti çözüm sunmayan, ancak mümkün olduğunca etkili çözümlerle gerçek hayattaki problemlere başarılı sonuçlar üreten algoritmalar içermektedir.



Şekil 7.1. Görsel olarak tipik bir optimizasyon süreci (Web Kaynağı 7.1)

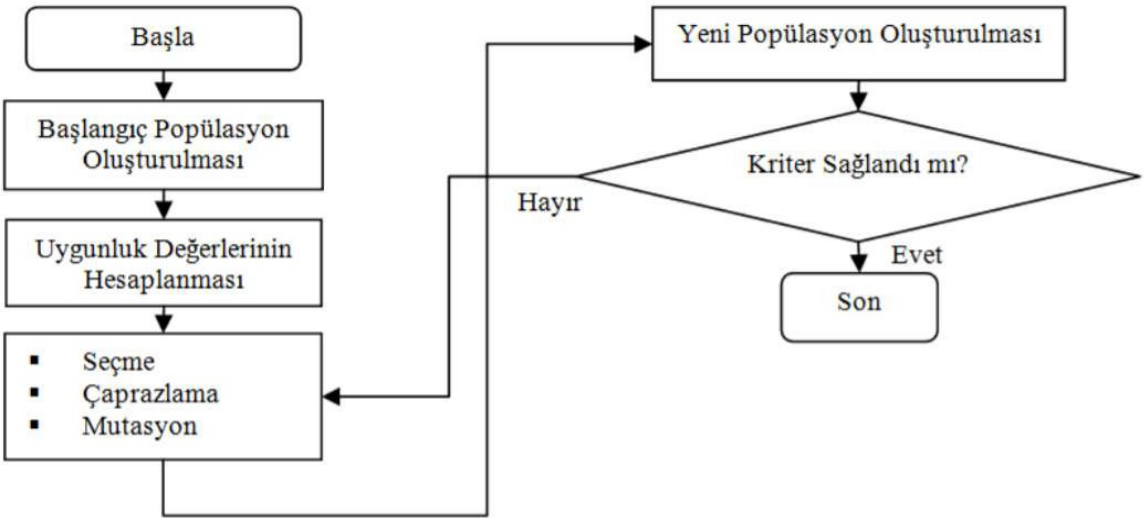
1.3. Genetik Algoritma ile Zeki Optimizasyon

Zeki optimizasyon konusunda yüzlerce algoritma bulunmaktadır. Ancak bunlardan bazıları konuyu anlamak ve başlangıç için daha idealdir. Genetik Algoritma da yapısı itibariyle buna en uygun algoritmalardan biridir. Genetik Algoritma, güçlü ve başarılı olan çözüm unsurlarının özelliklerinin yeni çözüm unsurlarına aktarılarak başarı şansının artırıldığı bir zeki optimizasyon algoritmasıdır. Buna göre mevcut popülasyonlardan (çözüm unsurlarından) problem fonksiyonu için nispeten daha iyi değer üretenler birbirleriyle eşleştirilerek daha başarılı yeni nesil bireylerden popülasyonlar elde edilmektedir. Hedeflenen döngü adımları boyunca nesillerin aynı şekilde türetilmesine devam edilmektedir. Genetik Algoritma orijinal yapısı itibariyle bir sürekli optimizasyon algoritmasıdır.

Genetik Algoritma'da bireylerin her biri problem fonksiyona ait aranılan değerleri tutmak adına gen adı verilen bileşenlerle tanımlanmaktadır. Genel olarak kodlama adı verilen bu süreç ile örneğin beş bilinmeyen değişkeni olan bir problem için *N* tane bireyin her birinde beşer adet değer / gen tutulmaktadır. Buradan hareketle her birey bir tür kromozom ile temsil edilir. Genetik Algoritma'nın her yeni döngüsünde, ilgili bireylerin taşıdığı değerler (genler) problem fonksiyonuna konularak fonksiyon sonuçları (uygunluk fonksiyonu değerleri) üretilmekte, belli sayıdaki en iyi değerleri bulan bireyler arası gen değişimleri (çaprazlama / crossover) ve belli genlerin de yeni değerlerle değiştirilmesi (mutasyon) suretiyle eski popülasyon yerine yeni nesil popülasyon (ebeveynlerin yerine çocuklar) elde edilerek bir sonraki döngüye geçilmektedir. Genetik Algoritma başarılı olan çözümleri gelecek süreçlere taşıyabilmesi nedeniyle Evrimsel Algoritmalar arasında kabul edilmektedir. Şekil 7.2 bu çerçevede Genetik Algoritma adımlarını kısaca görselleştirmektedir.

Eğitmene Not

Eğitmen öğrencilere Genetik Algoritma'daki kodlama, çaprazlama, çaprazlama için birey (elmen) eşleştirmeleri gibi süreçlerin birçok farklı şekilde yapılabildiğini açıklayarak, Genetik Algoritma'nın temellerindeki evrimsel süreçlere vurgu yapar. Ayrıca bu mekanizmaları canlılar arasındaki kalıtımsal özelliklerin aktarılmasına bağlayarak konunun pekişmesini sağlar.



Şekil 7.2. Genetik Algoritma'nın genel akışı (Web Kaynağı 7.2)

Düşün, tartış…

Doğadaki canlılar (örneğin kuşlar, arılar) problemlerini çözme konusunda ne şekillerde iş birlikleri yaparlar? Bunların zeki optimizasyona yansıması nasıl olabilir? Tartışalım!

1.4. Karınca Koloni Optimizasyonu ile Zeki Optimizasyon

Karınca Koloni Optimizasyonu, kombinasyonel optimizasyon söz konusu olduğunda, konuyu anlamak için en uygun algoritmadır. Söz konusu algoritma, karıncaların yiyecek kaynaklarına giden yolları birbirlerine işaret etmek için kullandıkları biyolojik mekanizmalara dayanan bir zeki optimizasyon algoritmasıdır. Buna göre karıncalar, gerçek dünyada yiyecek yollarını işaretlemek için feremon (pheremone) adı verilen bir madde salgılamaktadır. Bu yolla yiyeceklere giden en uygun yollar, sürüler için algılanır hale gelmektedir. Benzer şekilde kombinasyonal optimizasyon çözümleri için algoritma içerisinde tanımlanmış değişkenler birer feremon görevi görmekte, feremon değişken değerleri yüksek olan çözüm kombinasyonları bir fonksiyon üzerinden tespit edilebilmektedir.

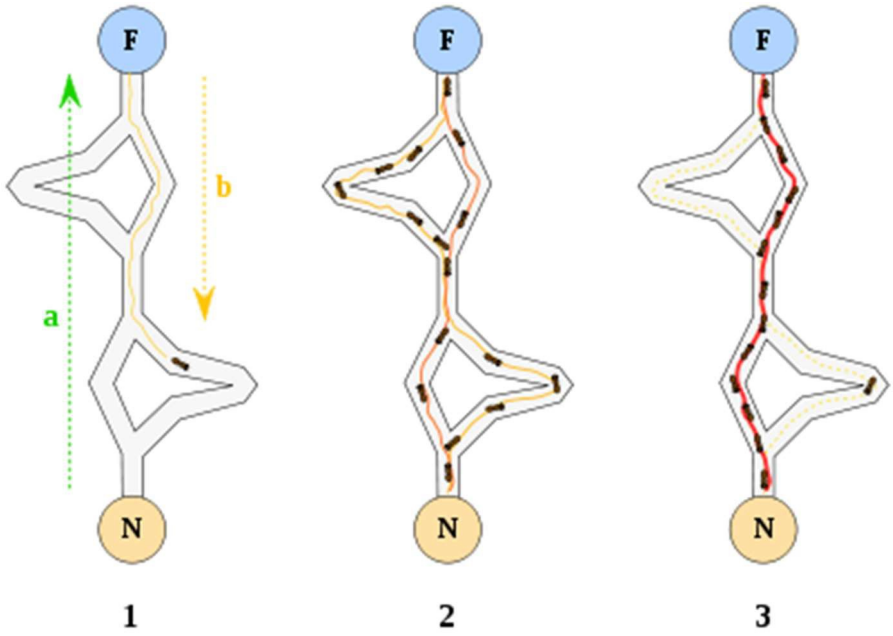
Eğitmene Not

Eğitmen öğrencilere kombinatöryal ve sürekli optimizasyon arasındaki farklılıkları, çözüm unsurları halihazırda bilinen; gezgin satıcı problemi (TSP: Travelling Salesman Problem) üzerinden örneklendirerek açıklar.

Karınca Koloni Optimizasyonunda karınca adı verilen her bir parçacık ve kurulan problem modeli, şu mekanizmalara dayanarak optimizasyon çözümünü desteklemektedir:

- *N* adet her bir karıncanın taşıdığı feremon değerleri vardır. Benzer şekilde, problem kapsamında aralarında potansiyel bağ olduğu düşünülen unsurlar da (örneğin en kısa yol bulunması istenen farklı şehirlerarası yollar) feremon değerleriyle temsil edilmektedir.
- Her döngü adımında karıncalar kombinasyonel adımlarla tespit ettikleri potansiyel unsurları problem fonksiyonundaki uygunluk / başarı değerini hesaplamak için kullanırlar.
- Uygun değer üreten karıncaların ve potansiyel çözüm unsurlarının feremon değerleri geliştirilir.
- Bir karıncanın aynı anda birden fazla potansiyel çözüm unsuruna yönelme durumu varsa, olasılıksal hesaplamalarla hangi tarafı tercih edeceği belirlenir.
- Algoritma tamamen çalışıp sona erdiğinde en çok feremon değeriyle desteklenen (en uygun) çözüm problemin çözümü olmaktadır.

Karınca Koloni Optimizasyonu'nun genel çözüm süreci Şekil 7.3'tekine benzer bir akışı ortaya çıkarmaktadır. Şekilde N ve F arası en uygun rota zamanla olgunlaşan; optimum çözümü oluşturmaktadır.



Şekil 7.3. Karınca Koloni Optimizasyonu ile temsili problem çözümü (Web Kaynağı 7.3)

Karınca Koloni Optimizasyonu kapsamında farklı potansiyel çözüm unsurları arası geçişlerde feremon değerlerinin güncellenmesi konusunda farklı matematiksel hesaplamalar gerçekleştirilmektedir. Bu hesaplamalar doğrultusunda karıncalar problem çözümünde tıpkı etmen tabanlı modellemede olduğu gibi yeni adımlar atabilmektedir.

Biliyor musunuz?

Zeki optimizasyon içerisinde yüzlerce farklı algoritma vardır. Araştırıp incelenebilecek alternatif bazı algoritmalar şu şekildedir:

- Parçacık Sürü Optimizasyonu
- Guguk Kuşu Algoritması
- Ateş Böceği Algoritması
- Yapay Arı Kolonisi Algoritması
- Girdap Optimizasyon Algoritması
- Akıllı Su Damlaları Algoritması
- Diferansiyel (Farklılaştırıcı) Evrim Algoritması

Simülasyon ile Zeki Optimizasyon

2. TASARLA

Genetik Algoritma kullanmak suretiyle bu paragraf altında verilen fonksiyonu 44 değerine eşitleyen x1, x2, x3, x4, x5, x6 değerlerinin bulunması istenmektedir (İlgili kodun tamamı Github platformu Hafta7 klasörü altında H7_genetik-algoritma.py dosyası ile sunulmuş durumdadır):

$$f(x1, x2, x3, x4, x5, x6) = 4x1 - 2x2 + 3{,}5x3 + 5x4 - 11x5 - 4{,}7x6$$

Öğrenciler, ilk olarak **pip install pygad** komutu ile Genetik Algoritma için kullanılan bir Python kütüphanesini kurarlar.

Eğitmen öğrencilerle birlikte Şekil 7.4'te gösterilen kodları yazar:

```
1 import pygad
2 import numpy
3
4 function_inputs = [4,-2,3.5,5,-11,-4.7]
5 desired_output = 44
6
7 def fitness_func(solution, solution_idx):
8     output = numpy.sum(solution*function_inputs)
9     fitness = 1.0 / numpy.abs(output - desired_output)
10    return fitness
11
12 fitness_function = fitness_func
13
14 num_generations = 100
15 num_parents_mating = 7
16
17 sol_per_pop = 50
18 num_genes = len(function_inputs)
19
20 last_fitness = 0
```

Şekil 7.4. Genel fonksiyon ve başlangıç parametrelerinin kodlanması

4. satırda fonksiyon değişken katsayıları tanımlanmakta, 5. satırda ise fonksiyonun vermesi istenen değer belirlenmektedir.

7.-10. satır arası fonksiyonun tanımlandığı Python kod bloğudur.

14. satırda optimizasyon toplam döngü sayısı 100 nesil sayısı ile tanımlanmaktadır.

15. satırda tanımlandığı üzere her döngüde en iyi 7 birey çaprazlama, mutasyon gibi işlemlerle yeni nesil popülasyon oluşturmak için seçilmektedir.

17. satırda tanımlandığı üzere, toplamda 50 adet bireyden oluşan bir popülasyon söz konusudur.

3. HAREKETE GEÇ

Problem ve algoritma tanımları sonrası her döngüde (nesilde) en iyi çözümleri ekrana yansıtan **nesil_ozeti** fonksiyon kod bloğu yazılır ve **pygad.GA** çağrısı ile Genetik Algoritma uygulaması tanımlanır (Şekil 7.5):

```
def nesil_ozeti(ga_instance):
    global last_fitness
    print("Nesil = {generation}".format(generation=ga_instance.generations_completed))
    print("Fonksiyon Sonucu = {fitness}".format(fitness=ga_instance.best_solution()[1]))
    print("Degisim = {change}".format(change=ga_instance.best_solution()[1] - last_fitness))
    last_fitness = ga_instance.best_solution()[1]

ga_instance = pygad.GA(num_generations=num_generations,
                       num_parents_mating=num_parents_mating,
                       fitness_func=fitness_function,
                       sol_per_pop=sol_per_pop,
                       num_genes=num_genes,
                       on_generation=nesil_ozeti)
```

Şekil 7.5. Genetik Algoritma çözümünün tanımlanması

4. YÜRÜT

Tanımlanan algoritma ve uygulama düzeni için **run** fonksiyonu ile optimizasyon süreci başlatılır. Optimizasyon sonucundaki en iyi / uygun değerler üç farklı değişken (en iyi çözüm değişken değerleri için **solution,** en iyi çözüm fonksiyonu sonucu için **solution_fitness** ve en iyi çözümü veren birey için **solution_idx**) altında toplanır:

```
ga_instance.run()

solution, solution_fitness, solution_idx = ga_instance.best_solution()
```

Şekil 7.6. En iyi çözüm tespitinin kodlanması

En uygun / iyi çözüme ilişkin değerler ekrana yazdırılır:

```
print("En uygun cozum degisken degerleri : {solution}".format(solution=solution))
print("En uygun cozumu veren birey indeks no.: {solution_idx}".format(solution_idx=solution_idx))

prediction = numpy.sum(numpy.array(function_inputs)*solution)
print("En uygun cozum ile fonksiyon sonucu : {prediction}".format(prediction=prediction))

if ga_instance.best_solution_generation != -1:
    print("En uygun cozum {best_solution_generation} nesil sonra elde edildi."
          .format(best_solution_generation=ga_instance.best_solution_generation))
```

Şekil 7.7. En iyi çözümün ekrana yazdırılması

5. KARAR VER

5.1. Sonuçların Gözlemlenmesi

Algoritma sürecinin başlatılması ile birlikte ekrana her döngüde / nesilde en uygun sonuca ilişkin bilgiler Şekil 7.8'de olduğu gibi yansıtılmaktadır:

```
Nesil = 86
Fonksiyon Sonucu = 903.3083029984443
Degisim = 0.0
Nesil = 87
Fonksiyon Sonucu = 903.3083029984443
Degisim = 0.0
Nesil = 88
Fonksiyon Sonucu = 903.3083029984443
Degisim = 0.0
Nesil = 89
Fonksiyon Sonucu = 903.3083029984443
Degisim = 0.0
```

Şekil 7.8. Nesillere göre en iyi çözümlerin ekrana yazdırılması

Ekrana yansıtılan sonuçlarda öncelikli olarak ilgili nesil sırası bilgisi, ardından o nesil içerisinde elde edilen en uygun değer (Fonksiyon Sonucu) yazdırılmaktadır. Bir önceki nesilde bulunan en iyi sonuç ile arada değişim olup olmadığı ise Degisim ibaresi altında gösterilmektedir.

Optimizasyon süreci bitmesiyle birlikte en iyi / uygun sonucu veren altı değişkenin değerleri, bu değerleri veren bireyin indeks numarası, fonksiyon sonucu ve en iyi sonucun ilk görülmeye başlandığı nesil sırası da ekranda belirtilmektedir (Şekil 7.9):

```
En uygun cozum degisken degerleri : [-1.16498213  1.98641117 -2.33159974  3.01034528
-3.70851693 -1.05251707]
En uygun cozumu veren birey indeks no.: 0
En uygun cozum ile fonksiyon sonucu : 43.99889295825503
En uygun cozum 27 nesil sonra elde edildi.
```

Şekil 7.9. Optimizasyon süreci sonucu çözüm

PYTHON KODLARI:

```
import pygad
import numpy

function_inputs = [4,-2,3.5,5,-11,-4.7]
desired_output = 44

def fitness_func(solution, solution_idx):
    output = numpy.sum(solution*function_inputs)
    fitness = 1.0 / numpy.abs(output - desired_output)
    return fitness
```

```
fitness_function = fitness_func

num_generations = 100
num_parents_mating = 7

sol_per_pop = 50
num_genes = len(function_inputs)

last_fitness = 0
def nesil_ozeti(ga_instance):
    global last_fitness
    print("Nesil =
{generation}".format(generation=ga_instance.generations_completed))
    print("Fonksiyon Sonucu =
{fitness}".format(fitness=ga_instance.best_solution()[1]))
    print("Degisim = {change}".format(change=ga_instance.best_solution()[1] -
last_fitness))
    last_fitness = ga_instance.best_solution()[1]

ga_instance = pygad.GA(num_generations=num_generations,
                       num_parents_mating=num_parents_mating,
                       fitness_func=fitness_function,
                       sol_per_pop=sol_per_pop,
                       num_genes=num_genes,
                       on_generation=nesil_ozeti)
ga_instance.run()
solution, solution_fitness, solution_idx = ga_instance.best_solution()
print("En uygun cozum degisken degerleri : {solution}".format(solution=solution))
print("En uygun cozumu veren birey indeks no.:
{solution_idx}".format(solution_idx=solution_idx))
```

```
prediction = numpy.sum(numpy.array(function_inputs)*solution)
print("En uygun cozum ile fonksiyon sonucu :
{prediction}".format(prediction=prediction))

if ga_instance.best_solution_generation != -1:
   print("En uygun cozum {best_solution_generation} nesil sonra elde edildi."
       .format(best_solution_generation=ga_instance.best_solution_generation))
```

Dünya'dan Haberler

Yapay zekâ, iklim değişikliğini tahmin edebilecek güçlü bir araç olabilir mi?

Yapay zekânın iddialı bir hedefi, Dünya'nın bir "dijital ikizi"ni, yani gezegenimizdeki sistemleri ve süreçleri taklit eden bir kopyasını oluşturmak. Dr. Mathieu, "*Bahsedilen 'dijital ikiz', dünyamızı yansıtan sayısal bir laboratuvar niteliğindedir. Burada çeşitli denemeler yapabilir, doğacak sonuçları değerlendirebilir ve edinilen bulgular ışığında gerekli politikaları oluşturabiliriz.*" diyor. BAS çevresel veri bilimcisi Dr. Scott Hosking, "*Doğal ortamların dijital ikizlerini geliştirmek için yapay zekâmız gerekli yapı taşlarına sahiptir ve bunlardan yola çıkarak nihayetinde Dünya'nın da bir dijital ikizini de oluşturabileceğiz.*" diye konuşuyor. "*Gezegenimizde değişen faktörlerin her birini gerekli detay seviyesinde izleyemiyoruz. Doğal ortamların dijital ikizlerini oluşturarak, kutup bölgeleri gibi erişimi zor olan bölgeler hakkında daha sağlıklı veriler oluşturabiliriz. Bu bilgiler, ölçüm yapacak insansız hava araçlarını ve denizaltılarını ölçüm yapmaları için daha hedefli olarak yönlendirebilmemizi sağlayabilir.*"

Fakat yapay zekâ hâlen kusursuz sonuçlar sunan bir teknoloji değil. Uzmanlar, elimizdeki verilerin iklim öngörülerinde bulunacak algoritmaları geliştirme konusunda yetersiz olduğuna dair uyarıda bulunuyor. Atmosferik ve Çevresel Araştırmalar (AER) mevsimsel tahmin direktörü ve MIT'de iklim bilimcisi olarak görev yapan Dr. Judah Cohen, bu konudaki görüşlerini şöyle ifade ediyor: "*Uyduların yaygınlaştığı 1979'dan beri toplanan verilerden faydalanıyoruz; fakat buna rağmen elimizde yapay zekâdan optimum sonuçlar alabilmemize yetecek kadar bilgi bulunmuyor. Belki oluşturduğumuz modeller temelinde yapay veriler oluşturabiliriz; fakat bu verilerin geçmişe dayalı gerçek veriler kadar sağlıklı olup olmayacağı konusu belirsiz.*" (Web Kaynağı 7.4).

6. İLAVE ETKİNLİK

Birbirine yollarla bağlı beş şehir arasında atılabilecek en kısa mesafeyi bulmak adına Karınca Koloni Optimizasyonu algoritması kullanılmak istenmektedir (Uygulamanın tamamı Github platformu Hafta7 klasörü altında iki dosya altında: H7_karinca-kolonisi.py ve H7_karinca-kolonisi_ornek.py dosyaları ile sunulmuş durumdadır).

Söz konusu probleme göre, Python ortamında farklı indekslerle temsil edilen şehirler arasındaki mesafe değerleri şu şekilde tanımlanmalıdır:

Çizelge 7.1. Şehirlerarasındaki mesafe değerlerinin tanımı.

	0	1	2	3	4
0	inf	2	2	5	7
1	2	inf	4	8	2
2	2	4	inf	1	3
3	5	8	1	inf	2
4	7	2	3	2	inf

Söz konusu kombinatöryal optimizasyon problemi gösteriminden yola çıkılarak Karınca Koloni Optimizasyonu algoritmasının öncelikli olarak temel parametrelerinin tanımlanması sağlanır:

```
1 import random as rn
2 import numpy as np
3 from numpy.random import choice as np_choice
4
5 class AntColony(object):
6
7     def __init__(self, distances, n_ants, n_best, n_iterations, decay, alpha=1, beta=1):
8         self.distances  = distances
9         self.pheromone = np.ones(self.distances.shape) / len(distances)
10        self.all_inds = range(len(distances))
11        self.n_ants = n_ants
12        self.n_best = n_best
13        self.n_iterations = n_iterations
14        self.decay = decay
15        self.alpha = alpha
16        self.beta = beta
17
```

Şekil 7.10. Karınca Koloni Optimizasyonu algoritmasının temel parametreleri

Ardından algoritmanın çalışmasını ve şehirlerarası yollara feremon değerlerinin atanmasını sağlayan fonksiyonlar, Şekil 7.11'de olduğu gibi tanımlanır.

```
18    def run(self):
19        shortest_path = None
20        all_time_shortest_path = ("placeholder", np.inf)
21        for i in range(self.n_iterations):
22            all_paths = self.gen_all_paths()
23            self.spread_pheronome(all_paths, self.n_best, shortest_path=shortest_path)
24            shortest_path = min(all_paths, key=lambda x: x[1])
25            print (shortest_path)
26            if shortest_path[1] < all_time_shortest_path[1]:
27                all_time_shortest_path = shortest_path
28            self.pheromone * self.decay
29        return all_time_shortest_path
30
31    def spread_pheronome(self, all_paths, n_best, shortest_path):
32        sorted_paths = sorted(all_paths, key=lambda x: x[1])
33        for path, dist in sorted_paths[:n_best]:
34            for move in path:
35                self.pheromone[move] += 1.0 / self.distances[move]
```

Şekil 7.11. Genel algoritma akış kodları

Her bir karıncanın geçtiği yolların mesafelerinin toplandığı ve yine her karıncanın geçtiği yolların kayıt altına alındığı fonksiyonlar kodlanır:

```
37    def gen_path_dist(self, path):
38        total_dist = 0
39        for ele in path:
40            total_dist += self.distances[ele]
41        return total_dist
42
43    def gen_all_paths(self):
44        all_paths = []
45        for i in range(self.n_ants):
46            path = self.gen_path(0)
47            all_paths.append((path, self.gen_path_dist(path)))
48        return all_paths
```

Şekil 7.12. Karıncaların yol değerlerinin kayıt altına alınması

Karıncaların daha önce ziyaret etmedikleri şehirlere uğramalarını ve bu sırada yol ayrımlarında feremon değerleri üzerinden olasılıksal olarak gidilebilecek şehrin seçilmesinin sağlandığı fonksiyonlar tanımlanır:

```
50    def gen_path(self, start):
51        path = []
52        visited = set()
53        visited.add(start)
54        prev = start
55        for i in range(len(self.distances) - 1):
56            move = self.pick_move(self.pheromone[prev], self.distances[prev], visited)
57            path.append((prev, move))
58            prev = move
59            visited.add(move)
60        path.append((prev, start))
61        return path
62
63    def pick_move(self, pheromone, dist, visited):
64        pheromone = np.copy(pheromone)
65        pheromone[list(visited)] = 0
66
67        row = pheromone ** self.alpha * (( 1.0 / dist) ** self.beta)
68
69        norm_row = row / row.sum()
70        move = np_choice(self.all_inds, 1, p=norm_row)[0]
71        return move
```

Şekil 7.13. Karıncaların olasılıksal şehir seçiminin kodlanması

Eğitmene Not

Eğitmen bu noktada öğrencilerden algoritma ana kod bloğunu ayrı bir dosya olarak kaydetmelerini ister ve yeni bir kod dosyası içerisinde problemin çözümleneceği kodların yazılmasını sağlar.

Yeni bir Python kod dosyası açıldıktan sonra, öncelikli olarak **numpy** ve kodlanan algoritmanın çağrısı yapılır. Ardından şehirlerarası mesafelerin tanımlandığı dizi veri yapısı tanımlanır:

```
1  import numpy as np
2
3  from ant_colony import AntColony
4
5  distances = np.array([[np.inf, 2, 2, 5, 7],
6                        [2, np.inf, 4, 8, 2],
7                        [2, 4, np.inf, 1, 3],
8                        [5, 8, 1, np.inf, 2],
9                        [7, 2, 3, 2, np.inf]])
10
```

Şekil 7.14. Şehir yapısının kodlanması

Problemin tanımı sonrasında toplamda 500 döngüde, 10 karınca üzerinden algoritma çağrısı yapılır:

```
11 ant_colony = AntColony(distances, 10, 1, 500, 0.95, alpha=0.05, beta=1)
12 shortest_path = ant_colony.run()
13 print ("En kısa yol: {}".format(shortest_path))
```

Şekil 7.15. Algoritma çağrısının kodlanması

Çağrı sonrasında **run** fonksiyonu ile başlatılan optimizasyon süreci sırasında her döngüde o ana denk bulunan en kısa gidiş rotası ekrana yazdırılır; döngüsel süreç sona erdiğinde ise en kısa mesafeyi veren nihai yol / rota ifade edilir:

```
([(0, 1), (1, 4), (4, 3), (3, 2), (2, 0)], 9.0)
([(0, 2), (2, 3), (3, 4), (4, 1), (1, 0)], 9.0)
([(0, 1), (1, 4), (4, 3), (3, 2), (2, 0)], 9.0)
([(0, 2), (2, 3), (3, 4), (4, 1), (1, 0)], 9.0)
([(0, 2), (2, 3), (3, 4), (4, 1), (1, 0)], 9.0)
([(0, 2), (2, 3), (3, 4), (4, 1), (1, 0)], 9.0)
([(0, 1), (1, 4), (4, 3), (3, 2), (2, 0)], 9.0)
([(0, 1), (1, 4), (4, 3), (3, 2), (2, 0)], 9.0)
En kısa yol: ([(0, 2), (2, 3), (3, 4), (4, 1), (1, 0)], 9.0)
```

Şekil 7.16. Genel algoritma sonuçlarının akışı

En kısa yol olarak, soldan sağa doğru hücre indeks değerleri dikkate alınmak suretiyle; örneğin (0, 2) ifadesi ile 0 indeksli şehirden 2 indeksli şehre, ardından (2, 3) ifadesi ile 2 indeksli şehirden de 3 indeksli şehre gidildiğini ve sürecin bu şekilde 9 birimlik mesafe ile en kısa yola tekabül ettiğini göstermektedir:

PYTHON KODLARI:

```
#karinca koloni algoritmasi kod bloğu
import random as rn
import numpy as np
from numpy.random import choice as np_choice


class AntColony(object):


   def _init_(self, distances, n_ants, n_best, n_iterations, decay, alpha=1,
beta=1):
      self.distances  = distances
      self.pheromone = np.ones(self.distances.shape) / len(distances)
      self.all_inds = range(len(distances))
      self.n_ants = n_ants
      self.n_best = n_best
      self.n_iterations = n_iterations
      self.decay = decay
```

```
        self.alpha = alpha
        self.beta = beta

    def run(self):
        shortest_path = None
        all_time_shortest_path = ("placeholder", np.inf)
        for i in range(self.n_iterations):
            all_paths = self.gen_all_paths()
            self.spread_pheronome(all_paths, self.n_best,
shortest_path=shortest_path)
            shortest_path = min(all_paths, key=lambda x: x[1])
            print (shortest_path)
            if shortest_path[1] < all_time_shortest_path[1]:
                all_time_shortest_path = shortest_path
            self.pheromone * self.decay
        return all_time_shortest_path

    def spread_pheronome(self, all_paths, n_best, shortest_path):
        sorted_paths = sorted(all_paths, key=lambda x: x[1])
        for path, dist in sorted_paths[:n_best]:
            for move in path:
                self.pheromone[move] += 1.0 / self.distances[move]

    def gen_path_dist(self, path):
        total_dist = 0
        for ele in path:
            total_dist += self.distances[ele]
        return total_dist

    def gen_all_paths(self):
        all_paths = []
```

```
    for i in range(self.n_ants):
        path = self.gen_path(0)
        all_paths.append((path, self.gen_path_dist(path)))
    return all_paths

def gen_path(self, start):
    path = []
    visited = set()
    visited.add(start)
    prev = start
    for i in range(len(self.distances) - 1):
        move = self.pick_move(self.pheromone[prev], self.distances[prev], visited)
        path.append((prev, move))
        prev = move
        visited.add(move)
    path.append((prev, start))
    return path

def pick_move(self, pheromone, dist, visited):
    pheromone = np.copy(pheromone)
    pheromone[list(visited)] = 0

    row = pheromone ** self.alpha * (( 1.0 / dist) ** self.beta)

    norm_row = row / row.sum()
    move = np_choice(self.all_inds, 1, p=norm_row)[0]
    return move
```

```
#örnek rota probleminin uygulanması
import numpy as np
from ant_colony import AntColony
distances = np.array([[np.inf, 2, 2, 5, 7],
                      [2, np.inf, 4, 8, 2],
                      [2, 4, np.inf, 1, 3],
                      [5, 8, 1, np.inf, 2],
                      [7, 2, 3, 2, np.inf]])
ant_colony = AntColony(distances, 10, 1, 500, 0.95, alpha=0.05, beta=1)
shortest_path = ant_colony.run()
print ("En kısa yol: {}".format(shortest_path))
```

Eğitmene Not

Eğitmen, ilgili hafta eğitimi hakkında öğrencilerin ilgisini arttırmak ve yeni öğrenilen bilgileri önceden öğrenilen bilgiler ile karşılaştırmak için aşağıda maddeler halinde verilen soruları tartışır.

- İlgili haftada ele alınan yapay zekâ uygulamalarına uygun örnek veri setleri neler olabilir?
- İlgili haftada ele alınan veri setlerinin yapay zekâ teknik ve kütüphanelerinde ne derecede başarım etkisi olmuştur?
- İlgili haftada öğretilen yapay zekâ tekniği ile uyumlu ve uyumsuz olabilecek açık erişimli veri seti çeşitleri sunan (kaagle, github vs.) internet sitelerinden elde edilecek farklı veri çeşitleri ile yapay zekâ tekniği uyumluluğunu sorar.

.

Kaynakça

Web Kaynağı 7.1:
https://www.researchgate.net/publication/320531598_Adaptive_optics_stochastic_optical_reconstruction_microscopy_AO-STORM_by_particle_swarm_optimization

Web Kaynağı 7.2: https://dergipark.org.tr/tr/download/article-file/353860

Web Kaynağı 7.3: https://www.biyologlar.com/karinca-surusu-optimasyonu

Web Kaynağı 7.4: https://tr.euronews.com/green/2020/10/12/bozulan-iklimimizi-yapay-zekâ-yard-m-yla-duzeltebilir-miyiz

8. Hafta: Derin Öğrenme ile İleri Düzey Çözümler

Ön Bilgi:

- Derin öğrenme kavramının temelleri, derin öğrenme modellemenin temelleri
- Python kodlamada döngüler, fonksiyonlar ve kütüphane yapılarının kullanımı (Bu konuda Python anlatım videoları: 14-16. Python kütüphaneleri serisi yardımcı araç olarak kullanılabilecektir).

Haftanın Kazanımları:

- Öğrenciler derin öğrenme model kavramını anlar.
- Öğrenciler, derin öğrenme modellerini büyük veri setlerine tasarlayarak uygular.
- Öğrenciler, derin öğrenme modellerinden elde edilen sonuçları yorumlayarak kavrar.
- Öğrenciler, derin öğrenme tabanlı problem çözebilme yetenekleri kazanır.
- Öğrenciler Python programlama dilinde derin öğrenme mimarileri için kullanılan kütüphane fonksiyonlarını kullanma yeteneği kazanır.
- Öğrenciler, Konvolüsyonel sinir ağları (CNN) derin öğrenme modelinin çalışma yöntemini kavrar.
- Öğrenciler CNN tabanlı AlexNet, ResNet, GoogleNet gibi mimarilerin yapısını kavrar.
- Öğrenciler ResNet tabanlı örnek bir problem üzerinde modeli tasarlar ve bu modeli çözüm yönünde uygular.
- Öğrenciler ResNet modelinden elde edilen sonuçları değerlendirerek yorumlar.
- Öğrenciler Yapay Zekâ alanı açısından derin öğrenme tekniğinin önemini kavrar.

Haftanın Amacı:

Bu haftanın amacı, “derin öğrenme (deep learning)” kavramının tüm öğrenciler tarafından doğru bir şekilde anlaşılmasını sağlamaktır. Haftanın bir diğer amacı, sıklıkla kullanılan derin öğrenme tabanlı ResNet modeli kullanılarak kedi köpek görüntüleri üzerinde sınıflandırma işlemini başarılı bir biçimde gerçekleştirmektir. Böylece, öğrencilerin Python programlama dilini kullanarak derin öğrenme tabanlı örnek problem modelleme ve çözüm üretme konusunda beceri kazanması sağlanacaktır.

Kullanılacak Malzemeler:

Bilgisayar, kâğıt, kalem, Python kod editörü, örnekler için çıktı görselleri

Haftanın İşlenişi:

Algıla: Derin öğrenme mimari kavramlarını öğrenir.

Tasarla: Derin öğrenme modelleme üzerinde kedi köpek görüntülerini ayırt etmek için çözüm süreçleri tasarlar.

Harekete Geç: Python ile derin öğrenme tabanlı bir ResNet mimari yapısı oluşturup uygular. Bu hafta için Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşim Harekete Geç aşaması ile başlatılabilmektedir.

Yürüt: Eğitmenler öğrenciler ile etkileşimli bir biçimde derin öğrenme tabanlı ResNet modelini çalıştırır ve genel çözüm akışını değerlendirir.

Karar Ver: Haftanın içeriğinin Github platformu (https://github.com/deneyapyz/ortaokul/) ile etkileşimli bir biçimde kullanır. Ayrıca öğrenciler, ResNet modelinden elde edilen sonuçları değerlendirerek tartışır.

1. ALGILA

1.1. Yapay Sinir Ağları Temelinde Derin Öğrenme'ye Geçiş

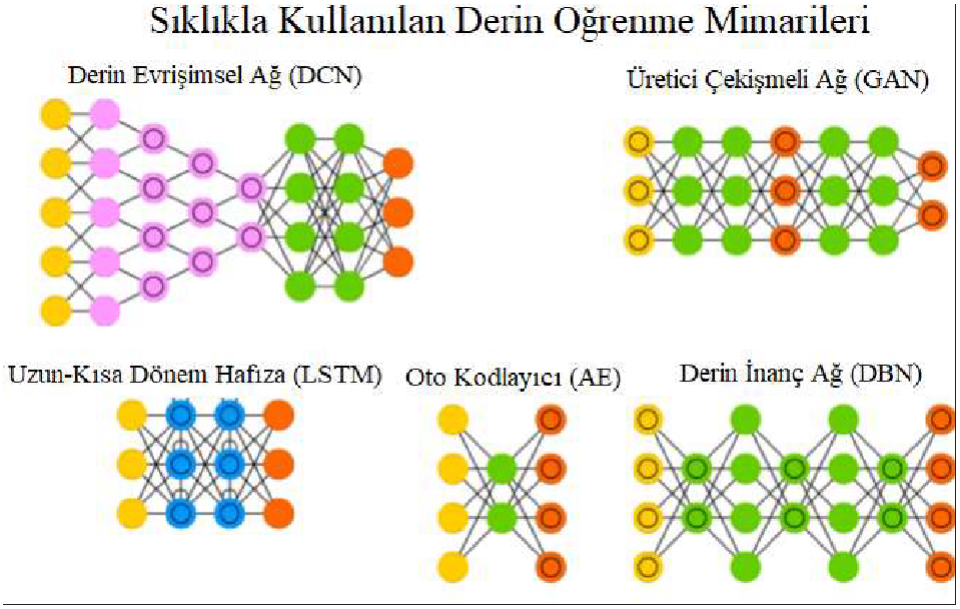
Eğitmene Not

Eğitmen, öğrencilere "**derin öğrenme**", "**derin öğrenme mimarisi**" ve "**derin öğrenmenin önemi**hakkında bilgi sahibi olup olmadıklarını sorar ve tartışır. Böylece öğrencilerin konuya dikkatini çeker.

Derin öğrenme, sayısal sistemlerin yapılandırılmamış, etiketlenmemiş verilere göre öğrenmesi ve kararlar alması için yapay sinir ağlarını kullanan makine öğrenmesinin bir alt dalıdır (Şekil 8.1). Yapay zekâ alanında çalışan uzmanlar, büyük veri olarak adlandırılan ve genellikle metin, ses veya resimlere dayalı veri kümelerini daha hızlı ve doğru bir şekilde analiz etmek için derin öğrenme mimarilerini kullanırlar.

Şekil 8.1. Yapay zekâ ve alt dalları

Yapay sinir ağlarındaki katman sayılarının artırılmasıyla kurulan çok farklı türde derin öğrenme mimarileri bulunmaktadır. Şekil 8.2'de sıklıkla kullanılan derin öğrenme mimarileri verilmiştir.



Şekil 8.2. Derin öğrenme mimarileri (Web Kaynağı 8.1)

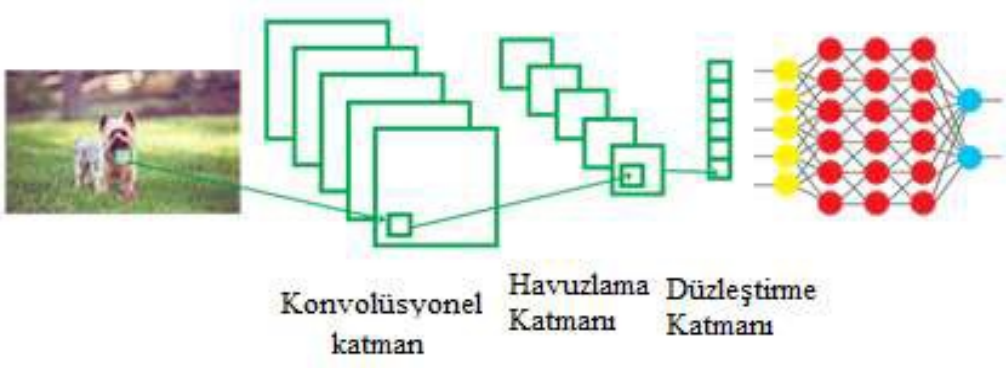
1.2. Evrişimsel Sinir Ağları Tekniği

Eğitmene Not

Eğitmen öğrencilere derin öğrenme mimarileri ve uygulama alanlarına ilişkin şu bilgileri aktararak konuyu pekiştirir:

Derin öğrenme mimarileri ile tıp, robotik, görüntü işleme, havacılık ve uzay sanayi, bilgisayarlı görü, görüntüden nesne tespiti, yüz tanıma, ses işleme-tanıma, finans gibi pek çok farklı alanda çözümler üretilmektedir (Doğan ve Türkoğlu., 2019).

Evrişimsel sinir ağı (**ConvNet / Convolutional neural networks -CNN**), girdi olarak alınan bir görüntüyü analiz ederek görünüşlerini veya görüntü içerisindeki görüntüleri birbirinden ayırt etmede sıklıkla kullanılan bir derin öğrenme mimarisidir. CNN derin öğrenme mimarisi Şekil 8.3'te görüldüğü gibi evrişim katmanı, doğrusal olmayan katman, havuzlama katmanı, düzleştirme katmanı ve tamamen bağlı katman yapılarından oluşmaktadır.



Şekil 8.3. CNN mimarisi (Web Kaynağı 8.2)

Eğitmene Not

Eğitmen öğrencilere CNN mimarilerinde kullanılan katmanların özelliklerini örnek vererek aktarır.

- **Konvolüsyonel Katman —** Hayvanın cinsi, ten rengi, şekli, çevre vb. fiziksel özellikleri saptamak için kullanılır.
- **Havuzlama Katmanı**— Havuzlama katmanı, görüntü üzerindeki boyutsallığı azaltmak için kullanılır. Böylece işlem sayısı azalırken, yakalanan gereksiz özellikler yok sayılır. Örneğin görüntüdeki köpeğin dışında kalan çevre gereksiz özellik olarak yok sayılır.
- **Düzleştirme Katmanı**— Tam bağımlı katmanda verileri analiz etmek için hazırlamada kullanılan katmandır. Örneğin önemli özellikleri verilen köpeğin görüntü halindeki iki boyutlu verilerini tek boyutlu veriye dönüştürerek tam bağımlı katmanda analiz edilmesini sağlar.
- **Tam Bağımlı Katman**— Tam bağımlı katman, CNN mimarisinin son ve en önemli katmanlardan birisidir. Tam bağımlı katman ile sınıflandırma veya regresyon işlemleri için eğitimler gerçekleştirilir. Örneğin görüntüden çıkartılan özelliklerin kedi köpek vs. gibi hangi hayvana ait olduğunu sınıflandırılmak için kullanılır.

Eğitmene Not

Eğitmen öğrencilere CNN tabanlı örnek mimariler hakkında şu derin öğrenme algoritmalarını örnek verip pekiştirerek aktarır:

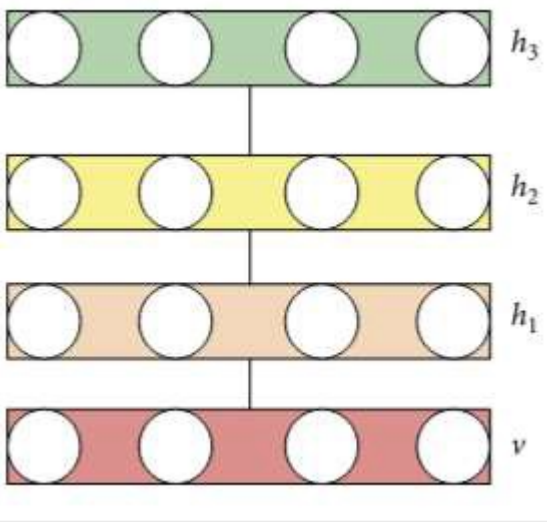
- **LeNet** — Bu ağ, Konvolüsyonel Sinir Ağlarının ilk başarılı uygulaması sayılır. 1990'lı yıllarda Yann LeCun tarafından geliştirilmiş ve posta kodlarını, basit basamakları, banka çeklerindeki rakamları vb. okumak için kullanılmıştır.
- **AlexNet —** Bu ağ, 2012'de Endüstri 4.0 devriminin önemli bileşenlerinden biri olan yapay zekânın bir alt bileşini olan derin öğrenmenin ve Konvolüsyonel sinir ağ modellerinin tekrar popüler hale gelmesini sağlamıştır. Alex Krizhevsky, Ilya Sutskever ve Geoffrey Hinton tarafından geliştirilmiştir. Uzaktan algılama, yabancı otların birbirinden ayırt edilmesi, araçların sınıflandırılması örneklerinde sıklıkla kullanılır.
- **GoogLeNet** —2014 yılında Google tarafından geliştirilen, hesaplama maliyetinin yüksek olduğu yapay zekâ uygulamalarında hız ve başarımı oldukça yüksek olan bir CNN mimarisidir. Yüz tanıma ve algılama sistemlerinde sıklıkla kullanılır.
- **VGGNet —** Bu ağ, VGG Group (Oxford) tarafından geliştirilmiştir. VGG mimarisi, AlexNet mimarisinde kullanılan yüksek işlevci (kernel) boyutlarının azaltılması ile oluşturulmuştur. VGG mimarisi konvolüsyonel sinir ağının başarımının arttırılması daha da derinleştirilmesi tekniği ile oluşturulmuştur. Nesne tanıma, beyin tümör tespiti, kuş türlerinin sınıflandırılması vb. gibi birçok alanda kullanılır.

Biliyor musunuz?

COVID-19 ile savaşmak için, derin öğrenmenin araştırmalara yardımcı olduğunu biliyor musunuz? En başarılı makine öğrenme algoritmalarından biri olan derin öğrenme için, biyoinformatik alanında DNA, RNA, protein dizileri ve Nanopore sinyali sekans verileri kaynak oluşturmaktadır. Elde edilen verilerde gözlenmiş veya daha önce bilinmeyen motifleri, modelleri ve alanları tespit etmede ve tanımlamada derin öğrenme kullanılır. Derin öğrenme ile; belirli hastalık ağları, gen birlikte ifade ağları, hücre sistemi hiyerarşi gibi grafik verileriyle başa çıkılabilir (Web Kaynağı 8.3).

1.3. Derin İnanç Ağları Tekniği

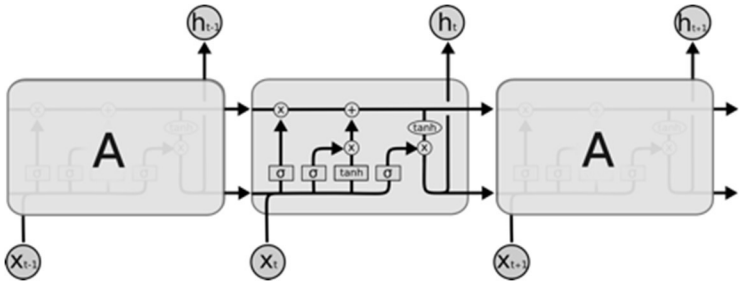
Derin İnanç Ağları (Deep Belief Network-DBN) genel olarak birden fazla gizli düğüm katmanından oluşan; düğümler yerine katmanlar arasında bağlantıların yer aldığı bir derin sinir ağı türüdür. Temel bir DBN mimarisinin yapısı Şekil 8.4'te verilmiştir.



Şekil 8.4. DBN model şeması (Dai et al., 2020).

1.4. Uzun-Kısa Dönem Hafıza Tekniği

Sepp Hochreiter ve Juergen Schmidhuber 1997 yılında vanishing gradient problemini çözmek için LSTM'i geliştirdiler. Daha sonra birçok kişinin katkısıyla düzenlenen ve popülerleşen LSTM şu anda geniş bir kullanım alanına sahiptir. LSTM tekniği, yeni bir kanal açma yoluyla sabit bir hata değeri sağlayarak recurrent ağların öğrenme adımlarının devam edebilmesini sağlamaktadır. Şekil 8.5'te LSTM tekniğinin ağ yapısı verilmiştir.

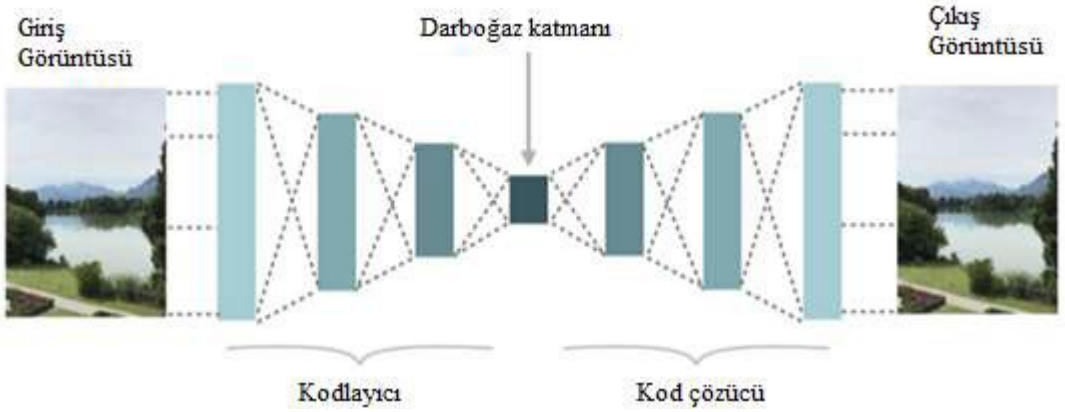


Şekil 8.5. LSTM Network Yapısı (Web Kaynağı 8.4)

LSTM yapısında tekrar eden modülün farkı tek bir yapay ağ katmanı yerine, özel bir şekilde bağlı kapı olarak adlandırılan dört katmandan oluşmasıdır. Normal akışın dışında dışarıdan bilgi alan bir yapıdır. Bu bilgiler depolanabilir, hücreye yazılabilir, okunabilir. Hücre neyi depolayacağına, ne zaman okumasına, yazmasına veya silmesine izin vereceğine kapılar sayesinde karar verir. Bu kapılarda bir ağ yapısı ve aktivasyon fonksiyonu bulunmaktadır. Aynı nöronlarda olduğu gibi gelen bilgiyi ağırlığına göre geçirir veya durdurur.

1.5. Oto kodlayıcı Tekniği

Oto kodlayıcı tekniği (auto encoder) derin öğrenme yöntemleri alanındaki en popüler modellerden birisidir. Oto kodlayıcı tekniği veriyi koda dönüştürerek otomatik olarak öğrenmeyi sağlar. Encoder (kodlayıcı) ve dekoder (kod çözücü) olarak iki kısımdan oluşur. Eğitim aşamasında tek bir modelmiş gibi beraber eğitilir (Şekil 8.6). Örneğin göndericiye kodlayıcı, alıcıya ise kod çözücü olarak çalışarak kişisel görüntülerimizin güvenli ve yüksek doğruluk oranında bir yerden başka bir yere taşınması sağlanabilir.



Şekil 8.6. Oto kodlayıcı tekniği şematiği (Web Kaynağı 8.5)

Biliyor musunuz?

Hayatımızı kolaylaştıran onlarca mobil uygulamaya ek olarak bazı uygulamaları çoğu zaman eğlenmek için kullanıyoruz. Yayına alındığı ilk 2 haftada 1 milyon iPhone kullanıcısıyla buluşan FaceApp uygulaması yapay sinir ağlarını makine öğrenmesi metodu ile bağdaştırıyor ve bu filtrelerle oldukça gerçek sonuçlar elde etmenizi sağlıyor. Facebook'un yüz tanıma özelliği, Apple cihazlarda bulunan Siri ve FaceApp gibi günlük hayatımızda sık sık kullandığımız uygulamalarını başarıya götüren Derin Öğrenme (Deep Learning) olduğunu biliyor muydunuz?

Ayrıca Snapchat, FaceApp, Instagram gibi yüz tanımaya dayalı katmanlar kullanan, Google görseller gibi Derin Öğrenme ile desteklenen uygulamalar önümüzdeki yıllarda içerik üretme konusunda işletmeler için oldukça büyük faydalar sağlayacak gibi görünüyor (Web Kaynağı 8.6)

2. TASARLA

Yapay zekâ görüntü tanımada başarılı sonuçlar sunuyor olsa da birbirine yakın özellikler içeren unsurları birbirine karıştırabilmektedir. Örneğin kedi ve köpek gibi, yakın özelliklere (ayak sayısı, kulak sayısı, dört ayak üzerinde yürümeleri, kuyruklarının olması…vs.) sahip hayvanları makine öğrenmesi algoritmaları karıştırabilmektedir. Neyse ki, derin öğrenme sayesinde bu sorun da aşılabilmektedir. Bu etkinlikteki veri setinde, kedi ve köpek olmak üzere iki sınıfa ait 25000 adet veri bulunmaktadır (Web Kaynağı 8.7). Söz konusu veri seti, bir tür Konvolüsyonel Sinir Ağı olan ResNet152_V2 derin öğrenme yöntemi kullanılarak sınıflandırılmıştır (Şekil 8.7).

Şekil 8.7. Veri setindeki örnek görüntü

Öğrenciler Çizelge 8.1'de verilen ResNet152_V2 derin öğrenme tekniği ile kedi köpek görüntü sınıflandırması için giriş çıkış parametrelerini belirler.

Çizelge 8.1. ResNet152_V2 derin öğrenme tekniği için giriş ve çıkış parametreleri

Veri Sayısı	GİRİŞ PARAMETRESİ	ÇIKIŞ PARAMETRESİ
	Kedi köpek görüntüsü	Sınıf
1		0 (Kedi)
1006		0(Kedi)
...	...	...
12500		0 (Kedi)
12501		1 (Köpek)

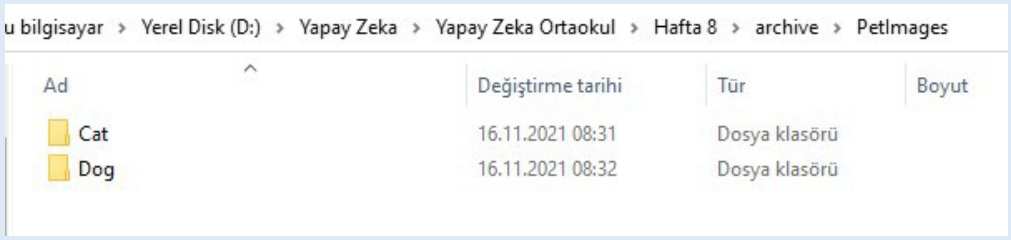
13025		1 (Köpek)
...	...	...
25000		1 (Köpek)

VERİ SETİ İÇİN İNDİRME LİNKİ

https://www.kaggle.com/shaunthesheep/microsoft-catsvsdogs-dataset

Eğitmene Not

Eğitmen, Şekil 8.8'de gösterildiği gibi öğrencilere veri setini anlatırken linkten indirilen verilerin Testing ve Training olarak ayrı bir şekilde indirildiğini anlatır.



Şekil 8.8. Veri seti klasörleri

3. HAREKETE GEÇ

Öğrenci ilgili veri seti dosyasını ResNet152_V2 derin öğrenme tekniği ile kedi köpek görüntüleri üzerinden hangisinin kedi hangisinin köpek olduğunu sınıflandırmaya yönelik işlemleri yapar (Uygulamanın tamamı Github platformunda, Hafta8 klasörü altındaki H8_derinogrenme_kedi-kopek.py kod dosyasıyla sunulmuştur. Uygulamanın veri seti ilgili Kaggle linki altından indirilmelidir.).

Öncelikli olarak, Şekil 8.9'da gösterildiği gibi, derin öğrenme eğitiminde kullanılacak kütüphane komutları yazılır.

1 -12 nolu kod satırında çalışma için gerekli kütüphaneleri çağırır. Burada 4, 9 ve 10 numaralı kod satırında derin öğrenme tekniği ile ilgili kütüphaneleri çağırır.

```
1   import matplotlib.pyplot as plt
2   import numpy as np
3   import seaborn as sns
4   import tensorflow as tf
5   from tqdm import tqdm
6   import os
7   from sklearn.metrics import classification_report,confusion_matrix
8   from PIL import Image
9   from tensorflow.keras.applications import ResNet152V2
10  from sklearn.model_selection import train_test_split
11  from warnings import filterwarnings
12  filterwarnings('ignore')
```

Şekil 8.9. Kuralların kod satırı

Öğrenciler, Şekil 8.10'da gösterilen 14-28 kod satırları arasında veri setinden veri okuma işlemini gerçekleştirir.

14. kod satırında; 'Cat' ve 'Dog' isminde iki adet sınıf tanımlar.

15. kod satırında; veri setinde yer alan görüntüleri 64 piksel boyutuna ayarlamak için **"image_size"** isminde bir değişken tanımlar.

16. ve 17. kod satırlarında; giriş ve çıkış eğitim verileri saklamak için **"X_train"** ve **"y_train"** isminde boş bir dizi oluşturur.

18. ile 28. Kod satırları arasında iç içe for döngüleri kullanılarak veri setindeki kedi köpek resimlerini 64 piksel olarak X_train değişkenine, bu resimlerin sınıflarını ise y_train değişkenine aktarır. "**try**", **"except"** yapısı kullanılarak veri setinde veri okumasında oluşabilecek problemlerin önüne geçilir.

```
14  labels = ['Cat','Dog']
15  image_size = 64
16  X_train = []
17  y_train = []
18  for i in labels:
19      folderPath = os.path.join('D:/Yapay Zeka/Yapay Zeka Ortaokul/Hafta 8/archive/PetImages/'+i)
20      klasor=folderPath+"/"
21      for j in tqdm(os.listdir(folderPath)):
22          yol=os.path.join(klasor,j)
23          try:
24              img=np.array(Image.open(yol).convert('RGB').resize((image_size, image_size), Image.ANTIALIAS))
25              X_train.append(img)
26              y_train.append(i)
27          except:
28              continue
```

Şekil 8.10. Veri setinden görüntülerin okunması

Öğrenciler, Şekil 8.11'de gösterilen 30-39 kod satırları arasında okunan görüntüler üzerinde eğitim yapmak için veri hazırlama işlemini gerçekleştirir.

30. ve 31. Kod satırlarında liste şeklinde saklanan giriş ve çıkış değişkenlerine ait verileri "**array"** formatına çevirir.

32. kod satırında veri setini eğitim verisi %90, test verisini %10 olacak şekilde rastgele olarak böler.

33. ve 37. Kod satırlarında kodlama isminde bir fonksiyon oluşturarak metin olarak verilen (kedi, köpek) ifadeleri sayısal değere (0, 1) çevrir.

38. ve 39. Kod satırlarında ise kodlama ismindeki fonksiyon çağırılarak sayısal çıkış değerleri y_train ve y_test değişkenlerine aktarır.

```
30  X_train=np.array(X_train)
31  y_train=np.array(y_train)
32  X_train,X_test,y_train,y_test = train_test_split(X_train,y_train, test_size=0.1,random_state=101)
33  def kodlama (y_t):
34      y_new = []
35      for i in y_t:
36          y_new.append(labels.index(i))
37      return tf.keras.utils.to_categorical(y_new)
38  y_train=kodlama(y_train)
39  y_test=kodlama(y_test)
40
```

Şekil 8.11. Görüntüler üzerinde eğitim yapmak için veri hazırlama kod satırları

Öğrenciler, Şekil 8.12'de gösterilen 41-47 kod satırları arasında ResNet modelini tanımlar.

41. nolu kod satırında ResNet152V2 modeli için başlangıç ağırlıklarını (weights) **'imagenet'** ten alarak, **"input_shape"** değişkeni ile görüntülerin boyutlarını belirleyerek tanımlar.

42. kod satırında ResNet152V2 mimarisinin çıkış değerini belirler.

43. nolu kod satırında ResNet152V2 mimarisinde bir adet havuzlama katmanı eklenmiştir.

44. nolu kod satırında ResNet152V2 mimarisinde %50 oranında "**Dropout"** katmanı eklenmiştir.

45. nolu kod satırında çıkış katmanı tanımlanarak aktivasyon fonksiyonu **"softmax"** belirlenmiştir.

46. nolu kod satırında ResNet152V2 mimarisinin giriş ve çıkış parametreleri belirlenmiştir.

47. nolu kod satırında ise oluşturulan ResNet152V2 mimarisinin özeti konsol ekranında görüntülenmiştir.

```
41  resnet = ResNet152V2(weights='imagenet', include_top=False,input_shape=(image_size,image_size,3))
42  model = resnet.output
43  model = tf.keras.layers.GlobalAveragePooling2D()(model)
44  model = tf.keras.layers.Dropout(rate=0.5)(model)
45  model = tf.keras.layers.Dense(2,activation='softmax')(model)
46  model = tf.keras.models.Model(inputs=resnet.input, outputs = model)
47  model.summary()
```

Şekil 8.12. ResNet152V2 mimarisinin oluşturulması için kod ekran görüntüsü

4. YÜRÜT

Öğrenciler kedi köpek görüntülerinin sınıflandırılması için oluşturmuş ResNet152V2 modelini eğitmek için Python kodlarını yazar. Şekil 8.13'te ekran görüntüsü verilmiştir.

49. nolu kod satırında modelin kayıp değerleri **"categorical_crossentropy",** optimizasyon yöntemi olarak **"adam"** ve değerlendirme metriği olarak da **"accuracy"** olarak belirler.

50. nolu kod satırında ResNet152V2 modelini **"X_train"** giriş ve **"y_train"** çıkış parametrelerine göre eğitir. Burada doğrulama verisini %10, epoch sayısını (eğitim sayısı) =5 ve her bir eğitimde alınacak olan veri sayısını (batch_size) 64 olarak belirler.

52. nolu kod satırında ResNet152V2 modelinde, eğitimden elde edilen sonuçların test verisi üzerinde doğruluğu test edilir.

53. ve 54. nolu kod satırlarında test verisinden elde edilen sonuçlara göre tahmin ve test verilerini değerlendirmek için aynı formata getirilmesi sağlanır.

```
49  model.compile(loss='categorical_crossentropy',optimizer = 'Adam', metrics= ['accuracy'])
50  model.fit(X_train,y_train,validation_split=0.1, epochs =5, verbose=1, batch_size=64)
51
52  pred = model.predict(X_test)
53  pred = np.argmax(pred,axis=1)
54  y_test_new = np.argmax(y_test,axis=1)
55
```

Şekil 8.13. ResNet152V2 modelinin eğitim ve tahmin için kod ekran görüntüsü

5. KARAR VER

Öğrenciler, kedi ve köpek olmak üzere ikili sınıfa ait derin öğrenme ResNet152V2 modelin sonuçlarına Şekil 8.14'te verilen kod satırlarını kullanarak karar verir.

56 nolu kod satırında test verilerinden elde edilen sınıflandırma sonuçlarını konsol ekranında görüntüler.

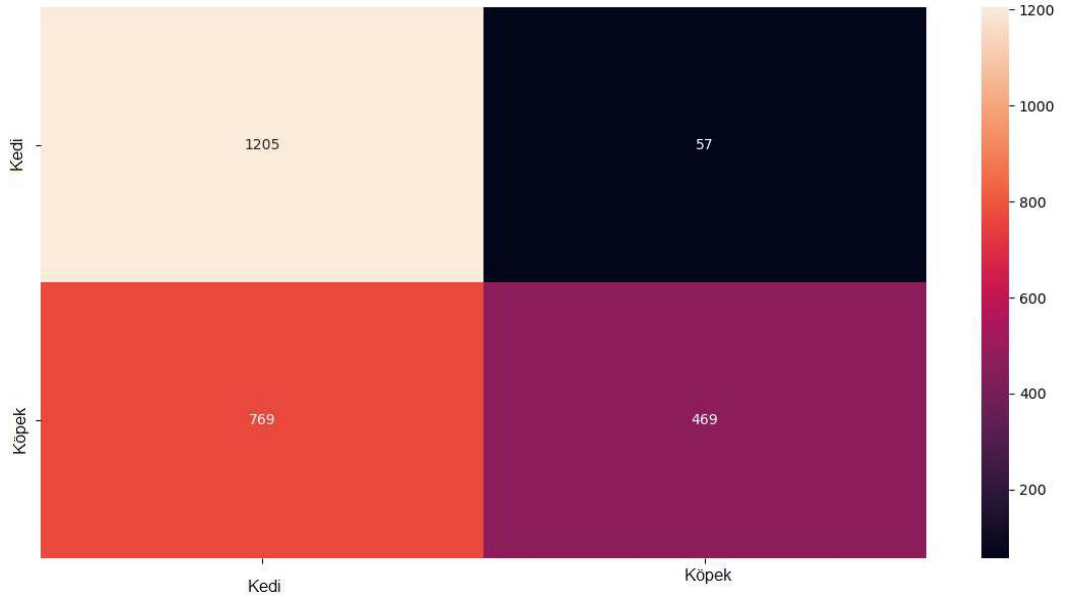
57–59 nolu kod satırlarında ise karmaşıklık matrisi sonuçlarını ekranda görüntüler.

Elde edilen karmaşıklık matrisi Şekil 8.15'te gösterilmiştir. Model doğruluğu %67 elde edilmiştir.

```
56  print(classification_report(y_test_new,pred))
57  fig,ax=plt.subplots(1,1,figsize=(14,7))
58  sns.heatmap(confusion_matrix(y_test_new,pred),ax=ax,xticklabels=labels,yticklabels=labels,annot=True)
59  plt.show()
```

Şekil 8.14. ResNet152V2 modelinden elde edilen sonuçların gösterimine ait kod ekran görüntüsü

	precision	recall	f1-score	support
0	0.61	0.95	0.74	1262
1	0.89	0.38	0.53	1238
accuracy			0.67	2500
macro avg	0.75	0.67	0.64	2500
weighted avg	0.75	0.67	0.64	2500



Şekil 8.15. Karmaşıklık matrisi görüntülemeleri

Eğitmene Not

Eğitmen, verilen kedi köpek görüntü sınıflarını eğitirken derin öğrenme ResNet152V2 mimarisindeki Python kodlarında, epochs ve batch_size değerlerini değiştirerek sonuçları öğrenciler ile tartışır.

Eğitmene Not

Eğitmen, öğrencilerle 1. Hafta'da Python yapay zekâ kütüphaneleri içerisinde hazır olarak bulunan **"İRİS ÇİÇEĞİ"** uygulamasını gerçekleştirirken, 3. Hafta'da Python yapay zekâ kütüphanelerinde hazır bir veri seti olarak bulunmayan açık erişimli internet sitesinden alınan **"ARAÇ ÖZELLİKLERİ"** "car.csv" dosyası üzerinden iki farklı yapay zekâ uygulaması gerçekleştirmiştir. Böylece hem Python içerisinde gömülü veri seti ile gömülü olmayan veri seti üzerinde uygulama yapma imkânı bulmuşlardır. 8. Haftada ise gömülü olmayan açık erişimli internet sitesinden alınan **büyük veri tabanlı bir veri seti** ile görüntülerin sınıflandırma işlemi gerçekleştirilmiştir.

6. UYGULAMANIN PYTHON KODLARI

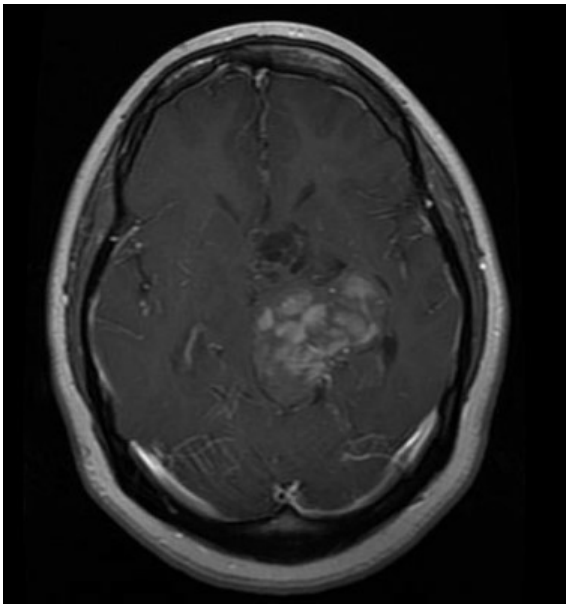
```
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import tensorflow as tf
from tqdm import tqdm
import os
from sklearn.metrics import classification_report,confusion_matrix
from PIL import Image
from tensorflow.keras.applications import ResNet152V2
from sklearn.model_selection import train_test_split
from warnings import filterwarnings
filterwarnings('ignore')

labels = ['Cat','Dog']
image_size = 64
X_train = []
y_train = []
for i in labels:
    folderPath = os.path.join('archive/PetImages/'+i)
    klasor=folderPath+"/"
    for j in tqdm(os.listdir(folderPath)):
        yol=os.path.join(klasor,j)
        try:
            img=np.array(Image.open(yol).convert('RGB').resize((image_size, image_size),
Image.ANTIALIAS))
            X_train.append(img)
            y_train.append(i)
        except:
            continue
```

```
X_train=np.array(X_train)
y_train=np.array(y_train)
X_train,X_test,y_train,y_test = train_test_split(X_train,y_train,
test_size=0.1,random_state=101)
def kodlama (y_t):
    y_new = []
    for i in y_t:
        y_new.append(labels.index(i))
    return tf.keras.utils.to_categorical(y_new)
y_train=kodlama(y_train)
y_test=kodlama(y_test)

resnet = ResNet152V2(weights='imagenet',
include_top=False,input_shape=(image_size,image_size,3))
model = resnet.output
model = tf.keras.layers.GlobalAveragePooling2D()(model)
model = tf.keras.layers.Dropout(rate=0.5)(model)
model   =   tf.keras.layers.Dense(2,activation='softmax')(model)
model = tf.keras.models.Model(inputs=resnet.input, outputs = model)
model.summary()
model.compile(loss='categorical_crossentropy',optimizer = 'Adam', metrics= ['accuracy'])
model.fit(X_train,y_train,validation_split=0.1, epochs =5, verbose=1, batch_size=64)

pred = model.predict(X_test)
pred = np.argmax(pred,axis=1)
y_test_new = np.argmax(y_test,axis=1)
print(classification_report(y_test_new,pred))
fig,ax=plt.subplots(1,1,figsize=(14,7))
sns.heatmap(confusion_matrix(y_test_new,pred),ax=ax,xticklabels=labels,yticklabels=labels,a
nnot=True)
plt.show()
```

7. İLAVE ETKİNLİK

Beyin tümörü, çocuklar ve yetişkinler arasında ölümcül hastalıklardan biri olarak kabul edilir. Her yıl yaklaşık 11.700 kişiye beyin tümörü teşhisi konulmaktadır. Beyin tümörü olan kişiler için genel olarak ortalama 5 yıllık bir yaşam süresi öngörülmektedir. Bu öngörü de erkeklerin yaklaşık yüzde 34'ü, kadınların ise yüzde 36'sı için yapılabilmektedir. Hastaların yaşam beklentisini iyileştirmek için doğru tedavi, planlama ve doğru teşhis uygulanmalıdır. Beyin tümörlerini saptamak için en iyi teknik Manyetik Rezonans Görüntüleme (MRI) tekniğidir. Her ne kadar MRI görüntüleri radyolog tarafından incelense de beyin tümörlerinde yer alan karmaşıklık düzeyi ve özellikleri nedeniyle manuel muayene genellikle hatalara neden olabilmektedir.

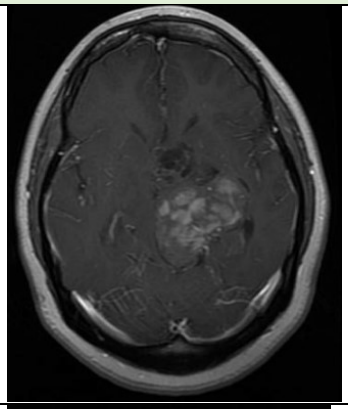
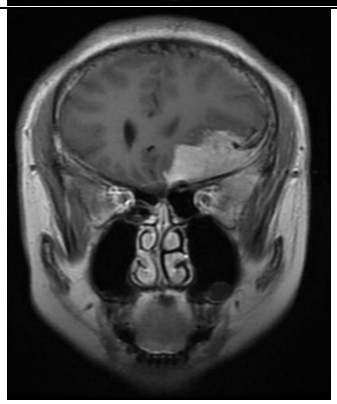
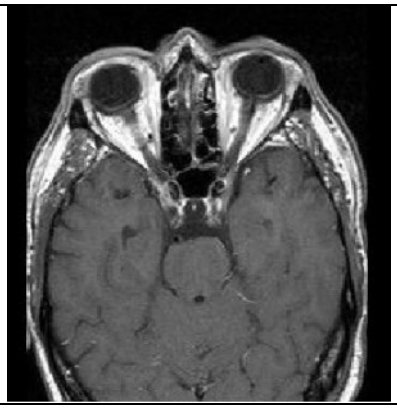
Derin öğrenme teknikleri kullanılarak insanlardan kaynaklanan hataları minimize etmek dünyanın her yerindeki doktorlara yardımcı olacaktır. Bu nedenle, açık erişimli internet sitesinden (Kaggle.com) alınan veri seti Konvolüsyonel Sinir Ağı olan ResNet152_V2 derin öğrenme yöntemi kullanılarak beyin tümörleri sınıflandırılmıştır (Şekil 8.16). İlgili uygulamanın kodu Github platformu Hafta8 klasöründe yer alan H8_derinogrenme_beyintumoru.py dosyası ile paylaşılmış durumdadır (Uygulamanın veri seti ilgili Kaggle linki altından indirilmelidir).

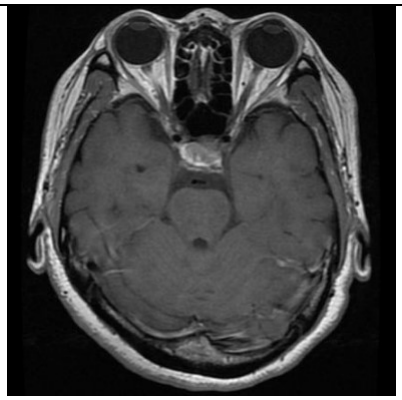


Şekil 8.16. MRI görüntüsü

Öğrenciler, Çizelge 8.2'de verilen ResNet152_V2 derin öğrenme tekniği ile beyin tümörü tespiti için giriş çıkış parametrelerini belirler.

Çizelge 8.2. ResNet152_V2 derin öğrenme tekniği için giriş ve çıkış parametreleri

Veri Sayısı	GİRİŞ PARAMETRESİ	ÇIKIŞ PARAMETRESİ
	MRI Görüntüsü	Tümör Sınıfı
1		Glioma_tumor
150		Meningioma_tumor
...	...	...
1440		No_tumor
...	...	...

3264			Pituitary_tumor

Bu bölümde hastalara ait 3264 adet veri setinde (Web Kaynağı 8.8). MRI görüntüsü giriş parametrelerine göre beyin tümörünün olup olmadığını ve türünü belirlemede ResNet152_V2 derin öğrenme tekniği ile sınıflandırılma yapılması amaçlanmıştır.

VERİ SETİ İÇİN İNDİRME LİNKİ

https://www.kaggle.com/sartajbhuvaji/brain-tumor-classification-mri

PYTHON KODLARI:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import cv2
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tqdm import tqdm
import os
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
from tensorflow.keras.applications import EfficientNetB0
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau, TensorBoard,
ModelCheckpoint
from sklearn.metrics import classification_report,confusion_matrix
import ipywidgets as widgets
```

```
import io
from PIL import Image
from IPython.display import display,clear_output
from warnings import filterwarnings
from PIL import Image

labels  =  ['glioma_tumor','no_tumor','meningioma_tumor','pituitary_tumor']

X_train = []
y_train = []
image_size = 128
for i in labels:
   folderPath = os.path.join('archive/','Training/',i)
   klasor=folderPath+"/"
   for j in tqdm(os.listdir(folderPath)):
      yol=os.path.join(klasor,j)
      try:
         img = Image.open(yol)
         img = img.resize((image_size, image_size))
         X_train.append(np.array(img))
         y_train.append(i)
      except:
         continue

for i in labels:
   folderPath = os.path.join('archive/','Testing/',i)
   klasor=folderPath+"/"
   for j in tqdm(os.listdir(folderPath)):
      yol=os.path.join(klasor,j)
      try:
         img = Image.open(yol)
```

```
            img = img.resize((image_size, image_size))
            X_train.append(np.array(img))
            y_train.append(i)
        except:
            continue

X_train = np.array(X_train)
y_train = np.array(y_train)

X_train, y_train = shuffle(X_train,y_train, random_state=101)

datagen = ImageDataGenerator(
    rotation_range=30,
    width_shift_range=0.1,
    height_shift_range=0.1,
    zoom_range=0.2,
    horizontal_flip=True)

datagen.fit(X_train)
print(X_train.shape)

X_train,X_test,y_train,y_test = train_test_split(X_train,y_train,
test_size=0.1,random_state=101)

y_train_new = []
for i in y_train:
    y_train_new.append(labels.index(i))
y_train = y_train_new
y_train  =  tf.keras.utils.to_categorical(y_train)

y_test_new = []
```

```
for i in y_test:
    y_test_new.append(labels.index(i))
y_test = y_test_new
y_test  =  tf.keras.utils.to_categorical(y_test)

from tensorflow.keras.applications import ResNet152V2
resnet = ResNet152V2(weights='imagenet',
include_top=False,input_shape=(image_size,image_size,3))

model = resnet.output
model = tf.keras.layers.GlobalAveragePooling2D()(model)
model = tf.keras.layers.Dropout(rate=0.5)(model)
model   =   tf.keras.layers.Dense(4,activation='softmax')(model)
model = tf.keras.models.Model(inputs=resnet.input, outputs = model)
model.summary()
model.compile(loss='categorical_crossentropy',optimizer  =  'Adam',  metrics=  ['accuracy'])

tensorboard = TensorBoard(log_dir = 'logs')
checkpoint =
ModelCheckpoint("resnet152v1.h5",monitor="val_accuracy",save_best_only=True,mode="auto
",verbose=1)
reduce_lr = ReduceLROnPlateau(monitor = 'val_accuracy', factor = 0.4, patience = 2,
                        min_delta = 0.001,mode='auto',verbose=1)

history = model.fit(X_train,y_train,validation_split=0.1, epochs =5, verbose=1, batch_size=64,
                callbacks=[tensorboard,checkpoint,reduce_lr])

filterwarnings('ignore')
epochs = [i for i in range(25)]
fig, ax = plt.subplots(1,2,figsize=(14,7))
train_acc = history.history['accuracy']
```

```
train_loss = history.history['loss']
val_acc = history.history['val_accuracy']
val_loss = history.history['val_loss']

colors_dark = ["#1F1F1F", "#313131", '#636363', '#AEAEAE', '#DADADA']
colors_red = ["#331313", "#582626", '#9E1717', '#D35151', '#E9B4B4']
colors_green = ['#01411C','#4B6F44','#4F7942','#74C365','#D0F0C0']

sns.palplot(colors_dark)
sns.palplot(colors_green)
sns.palplot(colors_red)

fig.text(s='Epochs vs. Training and Validation Accuracy/Loss',size=18,fontweight='bold',
         fontname='monospace',color=colors_dark[1],y=1,x=0.28,alpha=0.8)

sns.despine()
ax[0].plot(epochs, train_acc,
marker='o',markerfacecolor=colors_green[2],color=colors_green[3],
        label = 'Training Accuracy')
ax[0].plot(epochs, val_acc, marker='o',markerfacecolor=colors_red[2],color=colors_red[3],
        label = 'Validation Accuracy')
ax[0].legend(frameon=False)
ax[0].set_xlabel('Epochs')
ax[0].set_ylabel('Accuracy')

sns.despine()
ax[1].plot(epochs, train_loss,
marker='o',markerfacecolor=colors_green[2],color=colors_green[3],
        label ='Training Loss')
ax[1].plot(epochs, val_loss, marker='o',markerfacecolor=colors_red[2],color=colors_red[3],
        label = 'Validation Loss')
```

```
ax[1].legend(frameon=False)
ax[1].set_xlabel('Epochs')
ax[1].set_ylabel('Training & Validation Loss')

fig.show()
pred = model.predict(X_test)
pred = np.argmax(pred,axis=1)
y_test_new = np.argmax(y_test,axis=1)

print(classification_report(y_test_new,pred))

fig,ax=plt.subplots(1,1,figsize=(14,7))
sns.heatmap(confusion_matrix(y_test_new,pred),ax=ax,xticklabels=labels,yticklabels=labels,a
nnot=True,
        cmap=colors_green[::-1],alpha=0.7,linewidths=2,linecolor=colors_dark[3])
fig.text(s='Heatmap of the Confusion Matrix',size=18,fontweight='bold',
         fontname='monospace',color=colors_dark[1],y=0.92,x=0.28,alpha=0.8)
plt.show()
```

Eğitmene Not

Eğitmen, ilgili hafta eğitimi hakkında öğrencilerin ilgisini arttırmak ve yeni öğrenilen bilgileri önceden öğrenilen bilgiler ile karşılaştırmak için aşağıda maddeler halinde verilen soruları tartışır.

- İlgili haftada ele alınan yapay zekâ uygulamalarına uygun örnek veri setleri neler olabilir?
- İlgili haftada ele alınan veri setlerinin yapay zekâ teknik ve kütüphanelerinde ne derecede başarım etkisi olmuştur?
- İlgili haftada öğretilen yapay zekâ tekniği ile uyumlu ve uyumsuz olabilecek açık erişimli veri seti çeşitleri sunan (kaagle, github vs.) internet sitelerinden elde edilecek farklı veri çeşitleri ile yapay zekâ tekniği uyumluluğunu sorar.

Eğitmene Not

Eğitmen, öğrencilerin 7. ve 8. Hafta içerisinde elde ettikleri bilgi-becerileri değerlendirmek, aktif katılımlarını sağlamak için 'Kahoot' uygulamasını ya da benzeri bir Web 2.0 uygulamasını kullanarak 'bilgi yarışması' düzenleyebilir.

7. ve 8. Hafta bağlamında sorulabilecek sorular; zeki optimizasyon kavramının ve çözümlerinin temelleri, farklı zeki optimizasyon teknikleri ile uygulamalar, derin öğrenme kavramının temelleri, farklı derin öğrenme tekniklerinin temelleri ve ilgili çözüm yaklaşımları yönünde olabilir.

Kaynakça

Doğan, F., ve Türkoğlu, İ. (2019). Derin öğrenme modelleri ve uygulama alanlarına ilişkin bir derleme. Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi, 10(2), 409-445.

Dai, X., Cheng, J., Gao, Y., Guo, S., Yang, X., Xu, X., & Cen, Y. (2020). Deep belief network for feature extraction of urban artificial targets. Mathematical Problems in Engineering, 2020, Article ID 2387823.

Web Kaynağı 8.1: https://kim.hfg-karlsruhe.de/neural-network-chart/

Web Kaynağı 8.2: https://medium.com/@tuncerergin/convolutional-neural-network-convnet-yada-cnn-nedir-nasil-calisir-97a0f5d34cad

Web Kaynağı 8.3: https://www.bezelyedergi.net/post/biyoinformatik-derin-%C3%B6%C4%9Frenme-deep-learning

Web Kaynağı 8.4: https://medium.com/@hamzaerguder/recurrent-neural-network-nedir-bdd3d0839120

Web Kaynağı 8.5: https://www.bezelyedergi.net/post/biyoinformatik-derin-%C3%B6%C4%9Frenme-deep-learning

Web Kaynağı 8.6: https://blog.adresgezgini.com/faceapp-uygulamasiyla-yaslandik

Web Kaynağı 8.7: https://www.kaggle.com/shaunthesheep/microsoft-catsvsdogs-dataset

Web Kaynağı 8.8: https://www.kaggle.com/sartajbhuvaji/brain-tumor-classification-mri

Proje Yarışması

Yapay Zeka

Final Proje Yarışması / *Emisyon Sınıflarının Tahmini*

Eğitmene Not

Eğitmen öğrencilere önceki haftalarda genel hatlarıyla anlattığı; eğitim sonu proje yarışması hakkında detaylı bilgilendirmelerde bulunur. Eğitim süreci sonuna doğru, yarışmadaki probleme ilişkin olarak aşağıda sunulan bilgileri öğrenciler ile birlikte inceler ve irdeler.

Eğitmen aynı zamanda ilerleyen sayfalarda sunulan kuralları ve genel süreci dikkate alarak yarışma sürecinin gerçekleştirilip değerlendirilmesini sağlar. Eğitmenin dikkate alacağı kurallar ve değerlendirme dokümanları ayrıca sunulacaktır.

Sevgili Deneyap Öğrencimiz,

İklim değişikliği bütün dünyayı derinden etkileyen ve 20. yüzyıldan bu yana farklı sonuçlarla kendini hissettiren önemli bir küresel sorundur. İklim değişikliği sebepli sorunlar arasında anormal sıcaklıkların ve mevsim geçişlerinin görülmesi, doğal afetlerin artması, buzulların erimesi, ekosistemlerde dengelerin bozulması ve insanlarda kitlesel sağlık problemlerinin baş göstermesi yer almaktadır.

İklim değişikliği sorununa farklı insan faaliyetleri sebep olmakla birlikte özellikle küresel ısınma üzerindeki etkisi büyük olan sera gazı emisyonu, söz konusu sorunlar arasında oldukça kritik bir noktada yer almaktadır. Tıpkı diğer iklim değişikliği sorunları gibi sera gazı emisyonu sorununa da ülkemizdeki etkin çalışmalarla çözümler üretilmeye çalışılmaktadır.

Yapay Zekâ ile Tahmin Sistemi Tasarlayalım!

Deneyap Teknoloji Atölyeleri – Yapay Zekâ Dersi Final Projesi kapsamında, Birleşmiş Milletler tarafından sunulan, farklı ülkelerin 1990-2017 yılları arasındaki emisyon sınıflarını içeren ***'International Greenhouse Gas Emissions'* veri seti** üzerinde analizler yapan en başarılı yapay zekâ programını, derste öğrendikleriniz eşliğinde **Python kodlama dili** ile geliştirmeniz istenmektedir.

Veri setini buraya **tıklayarak** ulaşacağınız klasörden indirebilirsiniz.

Yarışma Kuralları

- Yarışma sadece Python kodu ile ders süreçlerinde anlatılan yapay zekâ tekniklerinin/yöntemlerinin kullanımını içermektedir. Farklı kodlama ortamları ve tekniklerle yazılan kodlar diskalifiye sebebidir.
- Yazılacak kod içerisinde sadece bir yapay zekâ tekniği kullanılması beklenmektedir. Birden fazla teknik kodu olması durumunda yukarıdan aşağıya doğru yazılan ilk teknik dikkate alınacaktır ancak başka teknikleri kullanmak da puan kazandıracaktır.
- Yarışma süreci 7. hafta sonundan 8. hafta dersi **2 gün öncesine kadar** sürmektedir. Kodlarınızı istediğiniz gibi düzenleyebilirsiniz. Ancak **Değerlendirme** başlığı altındaki kriterlere dikkat etmeniz yüksek puan almanızı sağlayacaktır.
- Hazırlamış olduğunuz programın **py** dosyasını **ekip_adı_il.py şeklinde isimlendirerek** (veri setine ayrıca gerek yoktur) **şu form linki** üzerinden, **8. hafta dersinden 2 gün öncesine kadar** iletiniz (Dosya iletmede sorun yaşarsanız, dosyayı deneyap.yz@gmail.com adresine ya da eğitmene e-posta yoluyla gönderebilirsiniz).
- **Veri seti *sınıflandırma problemine* tekabül etmektedir. Veri seti detaylarının anlaşılması ve kullanımı yarışmanın bir parçasıdır.**
- **Kodlayacağınız yapay zekâ algoritması veri setinin %50'si eğitim, %50'si test için kullanılmalıdır.**

Değerlendirme

Gönderilen her program (kod bütünü), önce aşağıdaki kıstaslara göre elde edilen **toplam puan** üzerinden değerlendirilecektir. **Her kriter 10 puandır.** Değerlendirmeler her ilin kendi içerisinde olacaktır:

Değerlendirme Kriteri	Kriter Onayı
Yapay zekâ kütüphanesinin çağırılması	**Evet / Hayır**
Veri setinin %50 eğitim %50 test şeklinde ayrılması	**Evet / Hayır**
Eğitim ve test veri setlerinin rastgele verilerle oluşturulması	**Evet / Hayır**
Yapay zekâ tekniğinin eğitilmesi	**Evet / Hayır**
Yapay zekâ tekniğinin (eğitimin) test edilmesi	**Evet / Hayır**
Bulguların ekrana doğruluk değeri ile yansıtılması	**Evet / Hayır**
Bulguların ekrana karmaşıklık matrisi ile yansıtılması	**Evet / Hayır**
Beş farklı çalıştırma sonunda ortalama %70'den yüksek doğruluk elde edilmesi	**Evet / Hayır**
Beş farklı çalıştırma sonunda ortalama %80'den yüksek doğruluk elde edilmesi	**Evet / Hayır**
Beş farklı çalıştırma sonunda ortalama %90'dan yüksek doğruluk elde edilmesi	**Evet / Hayır**
Aynı problem için başka yapay zekâ tekniklerinin (en az 2 farklı teknik) uygulanması	**Evet / Hayır**
Bütün yapay zekâ teknikleri ile elde edilen sonuçların karşılaştırılması	**Evet / Hayır**

Tüm öğrencilerimize başarılar dileriz.

DENEYAP TÜRKİYE
Yapay Zeka

YAPAY ZEKÂ

ORTAOKUL

DENEYAP TÜRKİYE