

ISTQB® Sertifikalı Test Uzmanı

Yapay Zekâ Testi

Ders Programı

Versiyon 1.0



International Software Testing Qualifications Board

Provided by

Alliance for Qualification, Artificial Intelligence United,
Chinese Software Testing Qualifications Board,
and Korean Software Testing Qualifications Board

A4Q

AiU Artificial Intelligence
United



Telif Hakkı Bildirimi

Telif Hakkı Bildirimi © International Software Testing Qualifications Board (bundan böyle ISTQB® olarak anılacaktır). ISTQB®, International Software Testing Qualifications Board'un tescilli ticari markasıdır.

Telif Hakkı © 2021 yazarları: Klaudia Dussa-Zieger (Başkan), Werner Henschelchen, Vipul Kocher, Qin Liu, Stuart Reid, Kyle Siemens ve Adam Leon Smith.

Tüm hakları saklıdır. Yazarlar, telif hakkını International ISTQB®'ye devretmektedir. Yazarlar (mevcut telif hakkı sahipleri olarak) ve ISTQB® (gelecekteki telif hakkı sahibi olarak) aşağıdaki kullanım koşullarını kabul etmişlerdir:

- Bu belgeden alınacak ve ticari kullanım amacı olmayan kısımlar, kaynağı belirtilmek şartıyla kopyalanabilir. Herhangi bir eğitim kurumu, yazarların ve ISTQB®'nin bu ders programının kaynağı ve telif hakkı sahibi olduklarını belirtmeleri durumunda, bu ders programını bir eğitim kursuna temel oluşturmak için kullanabilir; ancak böyle bir eğitim kursunun herhangi bir reklamı, yalnızca eğitim materyallerinin ISTQB® tarafından tanınan bir üye kurula (member board) resmi olarak akredite ettirilmesinden sonra yapılabilir.
- Herhangi bir kişi veya grup, yazarların ve ISTQB®'nin bu ders programının kaynağı ve telif hakkı sahibi olduğunu belirtmeleri şartıyla, bu ders programını makaleler ve kitaplar için temel olarak kullanabilir.
- ISTQB®'nin yazılı izni alınmadan bu ders programının başka herhangi bir şekilde kullanılması yasaktır.
- ISTQB® tarafından tanınan herhangi bir üye kurul, yukarıda belirtilen telif hakkı bildirimini ders programının tercüme edilmiş versiyonunda yeniden yayınlamak koşuluyla bu ders programını tercüme edebilir.

Revizyon GemiŖi

Versiyon	Tarih	Notlar
1.0	Ocak, 10, 2021	GA Sürümü
Türke eviri 1.0	Nisan 25, 2025	TTB® Sürümü

Teşekkür

ISTQB® Sertifikalı Test Uzmanı Temel Seviye Yapay Zekâ Testi Ders Programının Türkçeleştirme çalışmasına katkıda bulunan Yazılım Test ve Kalite Derneği çeviri çalışma grubu üyelerine burada tekrar teşekkür etmek isteriz. Çalışma grubu üyeleri (alfabetik sıraya göre):

- Ali Nurettin Demir
- Betül Ulu
- Deniz Onat
- Eda Civar
- Eda Erdem
- Esmâ Aslan
- Esra Karaarslan
- Furkan Dinçer
- Gülhanım Anulur
- Hilal Bayram Dartıcı
- Kadir Tepecik
- Kerim Anulur
- Koray Yitmen
- Meltem Aydaş
- Meltem Bayrak
- Özlemnur Bayram Dönmez
- Ramazan Yaşar
- Salih Topal
- Şermin Eldek
- Turgay Buğra Bayram
- Zeynep Soylu
- Yazılım Test ve Kalite Derneği, Nisan 2025

Yazılım Test ve Kalite Derneği Hakkında

(Turkish Testing Board – www.turkishtestingboard.org)

Yazılım Test ve Kalite Derneği, 2006 yılından bu yana Türkiye bilişim sektöründe yazılım testi farkındalığının artması ve gelişmesi için kâr amacı gütmeyen gönüllü bir şekilde aşağıdaki faaliyetleri gerçekleştirmektedir:

Uluslararası Sertifikasyon Sınavları

Dernek uluslararası ISTQB® sertifika sınavlarını gerçekleştirerek sınavlarda başarılı olan katılımcılara uluslararası geçerliliği olan sertifikalar vermektedir. 2006 yılından bu yana 10.000'den fazla test uzmanı adayı derneğe başvurarak sertifika sınavlarına girmiştir. Dernek bünyesinde Türkçe ve İngilizce olarak düzenlenmekte olan sertifika sınavları:

- ISTQB® Uluslararası Sertifikalı Temel Seviye Yazılım Test Uzmanı Sınavı
- ISTQB® Uluslararası Sertifikalı Temel Seviye Çevik Test Uzmanı Sınavı
- ISTQB® Uluslararası Sertifikalı İleri Seviye – Test Analisti Sınavı
- ISTQB® Uluslararası Sertifikalı İleri Seviye – Teknik Test Analisti Sınavı
- ISTQB® Uluslararası Sertifikalı İleri Seviye – Test Yöneticisi Sınavı
- ISTQB® Uluslararası Sertifikalı Test Otomasyon Mühendisi Sınavı
- ISTQB® Uluslararası Sertifikalı Performans Testi Sınavı
- ISTQB® Uluslararası Sertifikalı Yapay Zekâ Testi Sınavı
- ISTQB® Uluslararası Sertifikalı Mobil Uygulama Testi Sınavı
- ISTQB® Uluslararası Sertifikalı Otomotiv Yazılımı Testi Sınavı
- ISTQB® Uluslararası Sertifikalı Oyun Testi Sınavı

Uluslararası Testİstanbul Konferansları – www.testistanbul.org

Dernek, 2010 yılından bu yana Uluslararası Testİstanbul Konferanslarını düzenlemektedir. Geçtiğimiz 15 konferansta 50'den fazla ülkeden, 70'ten fazla konuşmacı ve 7.000'den fazla katılımcı ağırlanmıştır.

Paneller

Dernek, yazılım test sektörünün gelişimi için sektör veya konu bazlı paneller organize etmektedir. Bu panellere şu ana kadar 1.000'den fazla profesyonel katılım göstermiştir. Şimdiye kadar düzenlenen paneller:

- TestFintech,
- TestDefence,
- TestGames,
- TestInsurance,
- TestAnkara,
- Testİzmir,
- Test Finance,

Turkey Software Quality Report (TSQR)

Dernek tarafından 2011 yılından itibaren yüzlerce bilişim profesyoneli ve akademisyeninin katılımıyla her yıl düzenlenen anket sonuçlarının değerlendirilmesiyle hazırlanan, Türkiye bilişim sektörüne yön verir nitelikte çıkarımların olduğu rapordur. İngilizce yayınlanan rapor tüm ISTQB® üye dernekleri aracılığıyla 100'den fazla ülkedeki bilişim profesyoneline ulaşmaktadır.

ISTQB® Worldwide Software Testing Practices Report (WSTPR)

ISTQB® tarafından 100'den fazla ülkeden binlerce bilişim profesyoneli ve akademisyeninin katılımıyla düzenlenen anket sonuçlarının değerlendirilmesiyle hazırlanan, dünya bilişim sektörüne yön verir nitelikte çıkarımların olduğu rapordur.

Türkçeleştirme Çalışmaları

Uluslararası yazılım test terminolojisinin ülkemize kazandırılması için dernek bünyesinde yer alan gönüllü çeviri grubu ISTQB® dokümanlarının çevirisi üzerinde çalışmaktadır. Şu ana kadar çevrilen dokümanlar:

- ISTQB® Uluslararası Sertifikalı Temel Seviye Yazılım Test Uzmanı Ders Programı v4.0
- ISTQB® Uluslararası Sertifikalı Temel Seviye Yazılım Test Uzmanı Ders Programı 2018
- ISTQB® Uluslararası Sertifikalı Temel Seviye Yazılım Test Uzmanı Ders Programı 2011
- ISTQB® Yazılım Testi Terimler Sözlüğü
- ISTQB® Uluslararası Sertifikalı İleri Seviye – Test Analisti Ders Programı 2012
- Bir Ejderhadan Yazılım Test Dersi
- Çevik Bir Dünyada TMMi®
- ISTQB® Uluslararası Sertifikalı Mobil Uygulama Testi Ders Programı 2019
- TMMi® Genel Broşürü
- TMMi® Hızlı Tarama Aracı (Lightning Scan Tool) Dokümanı
- TMMi® Profesyonel Broşürü
- ISTQB Uluslararası Sertifikalı Temel Seviye Yazılım Test Uzmanı Örnek Sınav A

Burslar

Dernek her yıl kârından belli bir miktarı T.C. Üniversitelerinin Bilgisayar Mühendisliği, Yazılım Mühendisliği, Bilgisayar Programcılığı, Yönetim Bilişim Sistemleri ve bununla alakalı bölümlerinde okumakta olan başarılı ve ihtiyaç sahibi öğrencilere burs olarak aktarmaktadır. 2025 yılı itibarıyla burs sağlanan toplam bursiyer sayısı 100'ü geçmiştir.

İçindekiler Tablosu

Telif Hakkı Bildirimi.....	2
Revizyon Geçmişi	3
Teşekkür	4
Yazılım Test ve Kalite Derneği Hakkında	5
İçindekiler Tablosu	7
0. Giriş	12
0.1 Bu Ders Programının Amacı	12
0.2 YZ Testlerinde Sertifikalı Test Uzmanı	12
0.3 Sınav Kapsamındaki Öğrenme Hedefleri ve Bilginin Bilişsel Seviyesi	12
0.4 Uygulamalı Yeterlilik Seviyeleri	13
0.5 Sertifikalı Test Uzmanı YZ Testi Sınavı	13
0.6 Akreditasyon	14
0.7 Ayrıntı Düzeyi	14
0.8 Bu Ders Programı Nasıl Düzenlendi	14
1. Yapay Zekâya Giriş – 105 dakika	16
1.1 Yapay Zekâ Tanımı ve Etkisi	17
1.2 Dar, Genel ve Süper Yapay Zekâ (YZ).....	17
1.3 YZ Tabanlı ve Geleneksel Sistemler	17
1.4 YZ Teknolojileri	18
1.5 YZ Geliştirme Çerçevesi	18
1.6 YZ Tabanlı Sistemler için Donanım	19
1.7 YZ Servisleri (AlaaS)	20
1.7.1. YZ Servis Sözleşmeleri	20
1.7.2. Yapay Zekâ Servis Örnekleri (AlaaS)	20
1.8 Eğitim Öncesi Modeller	21
1.8.1. Eğitim Öncesi Modellere Giriş	21
1.8.2. Transfer Öğrenme	21
1.8.3. Eğitim Öncesi Modelleri ve Transfer Öğrenmeyi Kullanmanın Riskleri	22
1.9 Standartlar, Yönetmelikler ve YZ	22
2. YZ Tabanlı Sistemler için Kalite Özellikleri – 105 dakika	23
2.1 Esneklik ve Uyarlanabilirlik	24
2.2 Otonomi	24
2.3 Evrim	25

2.4 Yanlılık	25
2.5 Etik	26
2.6 Yan Etkiler ve Ödül Hileciliği	26
2.7 Şeffaflık, Yorumlanabilirlik ve Açıklanabilirlik	27
2.8 Güvenlik ve YZ	27
3. Makine Öğrenimi (MÖ) – Genel Bakış- 145 dakika	28
3.1 MÖ Çeşitleri	29
3.1.1. Denetimli Öğrenme	29
3.1.2. Denetimsiz Öğrenme	29
3.1.3. Takviyeli Öğrenme	30
3.2 MÖ İş Akışı	30
3.3 MÖ Türünü Seçme	34
3.4 MÖ Algoritması Seçiminde Yer Alan Faktörler	34
3.5 Aşırı Uyum ve Yetersiz Uyum	35
3.5.1. Aşırı Uyum	35
3.5.2. Yetersiz Uyum	35
3.5.3. Uygulamalı Egzersiz: Aşırı Uyum ve Yetersiz Uyumun Gösterilmesi	35
4. MÖ - Veri - 230 dakika	36
4.1 MÖ İş Akışının Bir Parçası Olarak Veri Hazırlama	37
4.1.1. Veri Hazırlığında Karşılaşılan Zorluklar	38
4.1.2. Uygulamalı Egzersiz: Makine Öğrenimi için Veri Hazırlığı	38
4.2 MÖ İş Akışında Eğitim, Doğrulama ve Test Veri Setleri	39
4.2.1. Uygulamalı Egzersiz: Eğitim ve Test Verilerini Tanımlama ve Bir MÖ Modeli Oluşturma	39
4.3 Veri Seti Kalite Sorunları.....	40
4.4 Veri Kalitesi ve MÖ Modeli Üzerindeki Etkisi	41
4.5 Denetimli Öğrenme için Veri Etiketleme	41
4.5.1. Veri Etiketleme Yaklaşımları	41
4.5.2. Veri Setlerinde Yanlış Etiketlenmiş Veriler	42
5. MÖ - Fonksiyonel Performans Metrikleri – 120 dakika	43
5.1 Karışıklık Matrisi	44
5.2 Sınıflandırma, Regresyon ve Kümeleme için Ek MÖ Fonksiyonel Performans Metrikleri	45
5.3 MÖ Fonksiyonel Performans Metriklerinin Sınırlamaları	45
5.4 MÖ Fonksiyonel Performans Metriklerinin Seçimi	46
5.4.1. Uygulamalı Egzersiz: Oluşturulan Makine Öğrenme (MÖ) Modelini Değerlendirme	47
5.5 MÖ için Kıyaslama Paketleri	47

6. MÖ -Yapay Sinir Ağları ve Test Edilmesi – 65 dakika	48
6.1 Yapay Sinir Ağları	49
6.1.1. Uygulamalı Egzersiz: Basit Bir Perseptron'u (Algılayıcı) Uygulama	50
6.2 Sinir Ağları için Kapsama Ölçüleri	51
7. YZ Tabanlı Sistemlerin Test Edilmesi Genel Bakışı– 115 dakika	52
7.1 YZ Tabanlı Sistemlerin Özellikleri	53
7.2 YZ Tabanlı Sistemler için Test Seviyeleri	53
7.2.1. Girdi Verisi Testi	54
7.2.2. MÖ Model Testi	54
7.2.3. Bileşen Testi	54
7.2.4. Bileşen Entegrasyon Testi.....	54
7.2.5. Sistem Testi.....	54
7.2.6. Kabul Testi	55
7.3 YZ Tabanlı Sistemlerin Test Edilmesi için Test Verileri	55
7.4 YZ Tabanlı Sistemlerde Otomasyon Yanlılığının Test Edilmesi	55
7.5 YZ Bileşenlerinin Belgelendirilmesi	56
7.6 Kavram Kayması İçin Test Etme	57
7.7 MÖ Sistemi için Bir Test Yaklaşımı Seçme	57
8. Yapay Zekâya Özgü Kalite Özelliklerini Test Etme– 150 dakika	60
8.1 Kendi Kendine Öğrenen Sistemleri Test Etmenin Zorlukları	61
8.2 Otonom Yapay Zekâ Tabanlı Sistemlerin Test Edilmesi	62
8.3 Algoritmik, Örnekleme ve Uygunsuz Yanlılık Testi	62
8.4 Olasılıksal ve Deterministik Olmayan YZ Tabanlı Sistemlerin Test Edilmesinde Karşılaşılan Zorluklar	63
8.5 Karmaşık YZ Tabanlı Sistemlerin Test Edilmesindeki Zorluklar	63
8.6 YZ Tabanlı Sistemlerin Şeffaflığının, Yorumlanabilirliğinin ve Açıklanabilirliğininin Test Edilmesi	64
8.6.1. Uygulamalı Egzersiz: Model Açıklanabilirliği	64
8.7 Yapay Zekâ Tabanlı Sistemler için Testin Doğruluk Ölçütleri	65
8.8 Test Amaçları ve Kabul Kriterleri	65
9. YZ Tabanlı Sistemlerin Test Edilmesi için Yöntem ve Teknikler – 245 dakika	68
9.1 Karşıt Saldırı ve Veri Zehirlenmesi	69
9.1.1. Karşıt Saldırı	69
9.1.2. Veri Zehirlenmesi	69
9.2 İkili Testler	70
9.2.1. Uygulamalı Egzersiz – İkili Testler	70
9.3 Sırt Sırt Test Etme	70

9.4 A/B Testi	71
9.5 Metamorfik Test (MT)	71
9.5.1. Uygulamalı Egzersiz: Metamorfik Test	73
9.6 YZ Tabanlı Sistemlerin Tecrübeye Dayalı Testi	73
9.6.1. Uygulamalı Egzersiz: Keşif Testi ve Keşifsel Veri Analizi (EDA)	75
9.7 YZ Tabanlı Sistemler için Test Tekniklerinin Seçilmesi	76
10. YZ Tabanlı Sistemler için Test Ortamları– 30 dakika	77
10.1 YZ Tabanlı Sistemler için Test Ortamları	78
10.2 YZ Tabanlı Sistemleri Test Etmek için Sanal Test Ortamları	78
11. YZ ile Test Etme – 195 dakika	80
11.1 Yazılım Testi için YZ Teknolojileri	81
11.1.1. Uygulamalı Egzersiz: Yapay Zekâ'nın Testlerde Kullanımı	81
11.2 Raporlanan Hataları Analiz Etmek için YZ Kullanımı	81
11.3 Test Senaryosu Oluşturma için YZ Kullanımı	82
11.4 Regresyon Test Gruplarının Optimizasyonu için YZ Kullanımı	82
11.5 Hata Tahmini için YZ Kullanımı	83
11.5.1. Uygulamalı Egzersiz : Bir Hata Tahmin Sistemi Kurma	83
11.6 Kullanıcı Arayüzlerini Test Etmek için YZ Kullanımı	83
11.6.1. Grafik Kullanıcı Arayüzünü (GUI) YZ ile Test Etme	83
11.6.2. Grafik Kullanıcı Arayüzü (GUI) 'nü Test Etmek için YZ Kullanımı	84
12. Referanslar	85
12.1 Standartlar [S].....	85
12.2 ISTQB® Dokümanları [I].....	85
12.3 Kitaplar ve Makaleler [B].....	86
12.4 Diğer Referanslar [R].....	88
13. Ek A - Kısaltmalar	90
14. Ek- B – YZ Özellikleri ve Diğer Terimler	91
15. Dizin	101

Teşekkür

Bu doküman ISTQB® Genel Kurulu tarafından 1 Ekim 2021'de resmi olarak yayımlanmıştır. Doküman ISTQB® bünyesinden bir ekip tarafından hazırlanmıştır: Klaudia Dussa-Zieger (başkan), Werner Henschelchen, Vipul Kocher, Qin Liu, Stuart Reid, Kyle Siemens ve Adam Leon Smith.

Ekip, katkıda bulunan üç müfredatın yazarlarına teşekkür eder;

- A4Q: Rex Black, Bruno Legeard, Jeremias Rößler, Adam Leon Smith, Stephan Goericke, Werner Henschelchen
- AiU: Baş yazarlar Vipul Kocher, Saurabh Bansal, Srinivas Padmanabhuni ve Sonika Bengani ve ortak yazarlar Rik Marselis, José M. Diaz Delgado
- KSTQB ve CSTQB AIT: Stuart Reid, Qin Liu, Yeoungjae Choi, Murian Song, Wonil Kwon

Ekip, müfredatın geliştirilmesi boyunca verdikleri destek için Sınav, Sözlük ve Pazarlama Çalışma Gruplarına, teknik düzenlemeleri için Graham Bath'a ve önerileri ve girdileri için Üye Kurullarına teşekkür eder. Bu müfredatın incelenmesine ve yorumlanmasına aşağıdaki kişiler katılmıştır: Laura Albert, Reto Armuzzi, Árpád Beszédes, Armin Born, Géza Bujdosó, Renzo Cerquozzi, Sudeep Chatterjee, Seunghee Choi, Young-jae Choi, Piet de Roo, Myriam Christener, Jean-Baptiste Crouigneau, Guofu Ding, Erwin Engelsma, Hongfei Fan, ter Földházi Jr., Tamás Gergely, Ferdinand Gramsamer, Attila Gyúri, Matthias Hamburg, Tobias Horn, Jarosław Hryzko, Beata Karpinska, Joan Killeen, Rik Kochuyt, Thomas Letzkus, Chunhui Li, Haiying Liu, Gary Mogyorodi, Rik Marselis, Imre Mészáros, Tetsu Nagata, Ingvar Nordström, Gábor Péterffy, Tal Pe'er, Ralph Pichler, Nishan Portoyan, Meile Posthuma, Adam Roman, Gerhard Runze, Andrew Rutz, Klaus Skafte, Mike Smith, Payal Sobti, Péter Sótér, Michael Stahl, Chris van Bael, Stephanie van Dijck, Robert Werkhoven, Paul Weymouth, Dong Xin, Ester Zabar, Claude Zhang.

0. Giriş

0.1 Bu Ders Programının Amacı

Bu ders programının amacı, ISTQB® Yapay Zekâ Testlerinde Sertifikalı Test Uzmanı'na yönelik bir çerçeve oluşturmaktır. ISTQB®, aşağıda belirtilen kişi ve kurumlarla bu ders programını paylaşır:

1. Üye Kurullar: Kendi dillerine çevirmeleri ve eğitim kurumlarını akredite etmeleri için. Üye Kurullar ders programını kendi özel dil ihtiyaçlarına göre uyarlayabilir ve yerel yayınlarına uyacak şekilde referansları değiştirebilir.
2. Sertifikasyon kurumları: Bu ders programının öğrenme hedeflerine uygun olacak şekilde kendi dillerinde sınav soruları hazırlamaları için.
3. Eğitim kurumları: Eğitim materyali üretmeleri ve uygun öğretim yöntemlerini belirlemeleri için.
4. Sertifika programı adayları: (bir eğitim kursunun parçası olarak veya bağımsız olarak) Sertifika sınavına hazırlanmaları için.
5. Uluslararası yazılım ve sistem mühendisliği topluluğu: Yazılım ve sistem test uzmanlığı mesleğini ilerletmek ve kitap ve makalelere bir temel oluşturmak için.

0.2 Yapay Zekâ Testlerinde Sertifikalı Test Uzmanı

Yapay Zekâ Testlerinde Sertifikalı Test Uzmanı, Yapay Zekâ tabanlı sistemleri ve/veya test için Yapay Zekâyla ilgilenen herkese yöneliktir. Buna, test uzmanları, test analistleri, veri analistleri, test mühendisleri, test danışmanları, test yöneticileri, kullanıcı kabul test uzmanları ve yazılım geliştiriciler gibi çeşitli rollerdeki kişiler dahildir. Bu sertifikasyon ayrıca, proje yöneticileri, kalite yöneticileri, yazılım geliştirme yöneticileri, iş analistleri, operasyon ekibi üyeleri, BT yöneticileri ve yönetim danışmanları gibi, Yapay Zekâ tabanlı sistemlerin test edilmesi ve/veya test için Yapay Zekânın temel anlayışını isteyen herkes için uygundur.

Yapay Zekâ Testlerinde Sertifikalı Test Uzmanı [I03], aşağıdaki bilgileri içeren ayrı bir belgedir:

- Ders programı için iş çıktıları
- İş sonuçlarının öğrenme hedefleri ile ilişkili matrisi
- Ders program özeti
- Ders programları arasındaki ilişkiler

0.3 Sınav Kapsamındaki Öğrenme Hedefleri ve Bilginin Bilişsel Seviyesi

Öğrenme hedefleri, iş çıktılarını destekler ve Sertifikalı Test Uzmanı YZ Testi sınavları oluşturmak için kullanılır.

Adaylardan herhangi bir on bir bölümde belirtilen bir anahtar kelimeyi veya kavramı hatırlamaları istenebilir. Özel öğrenme hedefi seviyeleri her bölümün başında gösterilir ve aşağıdaki şekilde sınıflandırılır:

- K1: Hatırla

- K2: Anla
- K3: Uygula
- K4: Analiz et

Konu başlıklarının hemen altında anahtar kelimeler olarak listelenen tüm terimlerin tanımlarının öğrenme hedeflerinde açıkça belirtilmiş olmasa bile hatırlanması gerekmektedir (K1).

0.4 Uygulamalı Yeterlilik Seviyeleri

Sertifikalı Test Uzmanı - Yapay Zekâ Testi, pratik beceriler ve yetkinliklere odaklanan uygulamalı hedefleri içerir.

Aşağıdaki seviyeler, uygulamalı hedefler için geçerlidir (gösterildiği gibi):

- H0: Bir egzersizin canlı gösterimi veya kayıtlı video.
- H1: Rehberli egzersiz. Öğrenciler, eğitmen tarafından gerçekleştirilen bir dizi adımı takip eder.
- H2: İpucu ile egzersiz. Öğrenciye, egzersizin belirli ipuçlarıyla birlikte verilmesi, böylece egzersiz belirli bir süre içinde çözülebilir veya öğrencilerin bir tartışmaya katılması sağlanır.

Yetkinlikler, aşağıdaki listede gösterildiği gibi uygulamalı egzersizler yaparak elde edilir:

- Yetersiz uyum ve aşırı uyum göstermek (H0).
- Bir Makine Öğrenimi (MÖ) modelinin oluşturulmasını desteklemek için veri hazırlığı yapmak (H2).
- Eğitim ve test veri setlerini tanımlamak ve bir Makine Öğrenimi (MÖ) modeli oluşturmak (H2).
- Oluşturulan Makine Öğrenimi (MÖ) modelini seçilen Makine Öğrenimi (MÖ) işlevsel performans metrikleri kullanarak değerlendirmek (H2).
- Bir perseptron'un (Algılayıcı) uygulanması deneyimi (H1).
- Test uzmanları tarafından açıklanabilirliğin nasıl kullanılabileceğini göstermek için bir araç kullanmak (H2).
- Bir Yapay Zekâ tabanlı sistem için çiftli testi uygulamak ve test vakaları türetmek ve yürütmek (H2).
- Belirli bir senaryo için test vakalarını türetmek ve yürütmek için metamorfik test uygulamak (H2).
- Bir Yapay Zekâ tabanlı sistem üzerinde keşif testi uygulamak (H2).
- Yapay Zekâ kullanılmayan test etkinliklerini örneklerle tartışmak (H2).
- Basit bir Yapay Zekâ tabanlı hata tahmin sistemi uygulamak (H2).

0.5 Sertifikalı Test Uzmanı Yapay Zekâ Testi Sınavı

Sertifikalı Test uzmanı YZ Testi sınavı bu müfredata dayanmaktadır. Sınav sorularını cevaplamak için, bu ders programının birden fazla bölümünü baz alan ders materyali gerekebilir. Giriş ve ekler hariç olmak üzere bu ders programının tüm bölümleri sınav kapsamındadır. Bu ders programına bazı standartlar ve kitaplar referans olarak dahil edilmiştir ancak bunların içerikleri, bu ders programında bahsedildiği kadar sınava dahildir. Bahsedilmeyen kısım ve içerikler sınav kapsamına dahil değildir.

Daha fazla ayrıntı için Sertifikalı Test Uzmanı - Yapay Zekâ Testi "Genel Bakış" belgesine başvurun.

Giriş Koşulu Notu: Sertifikalı Test Uzmanı YZ Testi sınavına girmeden önce ISTQB® Temel Seviye sertifikası alınmalıdır.

0.6 Akreditasyon

Bir ISTQB® Üye Kurulu, ders materyali bu ders programına uygun olan eğitim kurumlarını akredite edebilir. Eğitim kurumları, akreditasyonu gerçekleştiren üye kuruldun veya kuruluştan akreditasyon rehberini edinmelidir. Akredite edilmiş bir kursun bu ders programına uygun olduğu kabul edilir ve kursun bir parçası olarak ISTQB® sınavına girilmesine izin verilir.

Bu ders programının akreditasyon yönergeleri Süreç Yönetimi ve Uyumluluk Çalışma Grubu tarafından yayınlanan genel Akreditasyon Yönergelerine uygundur.

0.7 Ayrıntı Düzeyi

Bu ders programındaki ayrıntı düzeyi, uluslararası düzeyde tutarlı kurs ve sınavlara olanak sağlar. Bu amaca ulaşmak için, ders programında aşağıdakilere yer verilmiştir.

- Sertifikalı Yazılım Test Uzmanı YZ Testi'nin niyetini açıklayan genel öğretici hedefler.
- Öğrencilerin hatırlaması gereken terimlerin listesi.
- Her bilgi alanı için öğrenme ve uygulamalı hedefler, ulaşılması gereken öğrenme sonuçlarını tanımlar.
- Kaynaklar, kabul edilmiş literatür veya standartlar gibi referanslara atıfta bulunularak ana kavramların açıklaması.

Ders programı içeriği, Yapay Zekâ tabanlı sistemlerin testi için tüm bilgi birikiminin bir açıklaması değildir; bu, Sertifikalı Yazılım Test Uzmanı YZ Testi eğitim kurslarında ele alınacak ayrıntı düzeyini gösterir. Temel Yapay Zekâ (YZ) kavramlarını ve özellikle Makine Öğrenimi kavramlarını tanıtmaya odaklanır ve bu teknolojilere dayalı sistemlerin nasıl test edilebileceğini inceler.

0.8 Bu Ders Programı Nasıl Düzenlendi

Ders programı sınava tabi olan on bir konudan oluşmaktadır. Her konuda yer alan en üstteki başlık o konu için ayrılan eğitim süresini belirtir. Zaman planlaması o konunun geneli için verilmiştir, daha detayda bir planlama verilmemiştir. Akredite eğitim kursları için ders programı aşağıdaki gibi on bir konuya ayrılmış olup toplamda en az 25,1 saatlik kurs alınmasını gerektirir:

- Bölüm 1: 105 dakika Yapay Zekâ'ya Giriş
- Bölüm 2: 105 dakika Yapay Zekâ Tabanlı Sistemler için Kalite Özellikleri
- Bölüm 3: 145 dakika Makine Öğrenimi (MÖ) - Genel Bakış
- Bölüm 4: 230 dakika MÖ - Veri
- Bölüm 5: 120 dakika MÖ İşlevsel Performans Metrikleri
- Bölüm 6: 65 dakika MÖ - Sinir Ağları ve Test

- Bölüm 7: 115 dakika Yapay Zekâ Tabanlı Sistemlerin Test Edilmesi Genel Bakış
- Bölüm 8: 150 dakika Yapay Zekâ Özgü Kalite Özelliklerinin Test Edilmesi
- Bölüm 9: 245 dakika Yapay Zekâ Tabanlı Sistemlerin Testi için Yöntemler ve Teknikler
- Bölüm 10: 30 dakika Yapay Zekâ Tabanlı Sistemler için Test Ortamları
- Bölüm 11: 195 dakika Test İçin Yapay Zekâ Kullanımı

1.Yapay Zekâ'ya Giriş-105 Dakika

Test Edilen Anahtar Kelimeler

Yok

YZ-Spesifik Anahtar Kelimeleri

Servis Olarak Yapay Zekâ (AlaaS), Yapay Zekâ Geliştirme Çerçevesi, Yapay Zekâ Etkisi, Yapay Zekâ Tabanlı Sistem, Yapay Zekâ (YZ), Nöral Ağ, Derin Öğrenme (DÖ), Derin Nöral Ağı, Genel Yapay Zekâ, Genel Veri Koruma Yönetmeliği (GDPR), Makine Öğrenimi (MÖ), Dar Yapay Zekâ, Önceden Eğitilmiş Model, Süper Yapay Zekâ, Teknolojik Tekillik, Transfer Öğrenme

Bölüm 1 - Öğrenme Hedefleri:

1.1 YZ Tanımı ve YZ Etkisi

YZ-1.1.1 K2 Yapay Zekâ etkisini ve bunun Yapay Zekânın tanımını nasıl etkilediğini açıklayın.

1.2 Dar, Genel ve Süper Yapay Zekâ

YZ-1.2.1 K2 Dar Yapay Zekâ, Genel Yapay Zekâ ve Süper Yapay Zekâ arasındaki farkları belirtin.

1.3 Yapay Zekâ Tabanlı ve Geleneksel Sistemler

YZ-1.3.1 K2 Yapay Zekâ tabanlı sistemler ile geleneksel sistemler arasındaki farkları belirtin.

1.4 Yapay Zekâ Teknolojileri

YZ-1.4.1 K1 Yapay Zekâ'yı uygulamak için kullanılan farklı teknolojileri tanıyın.

1.5 Yapay Zekâ (YZ) Geliştirme Çerçevesi

YZ-1.5.1 K1 Popüler Yapay Zekâ geliştirme çerçevelerini tanımlayın.

1.6 Yapay Zekâ Tabanlı Sistemler için Donanım

YZ-1.6.1 K2 Yapay Zekâ tabanlı sistemleri uygulamak için mevcut donanım seçeneklerini karşılaştırın.

1.7 Servis olarak Yapay Zekâ (AlaaS)

YZ-1.7.1 K2 YZ Servis olarak Yapay Zekâ (AlaaS) kavramını açıklayın.

1.8 Önceden Eğitilmiş Modeller

YZ-1.8.1 K2 Önceden eğitilmiş Yapay Zekâ modellerinin kullanımını ve bunlarla ilişkili riskleri açıklayın.

1.9 Standartlar, Yönetmelikler ve Yapay Zekâ (YZ)

YZ-1.9.1 K2 Yapay Zekâ tabanlı sistemlere nasıl standartların uygulandığını açıklayın.

1.1 Yapay Zekâ (YZ) Tanımı ve Etkisi

Yapay Zekâ (YZ) terimi 1950'lere dayanmakta ve insanları taklit edebilen "akıllı" makineleri inşa etme ve programlama hedefine atıfta bulunmaktadır. Bugün için tanım önemli ölçüde evrim geçirmiştir ve aşağıdaki tanım, kavramı yakalamaktadır [S01]:

Mühendislikle oluşturulmuş bir sistemin bilgi ve becerileri edinme, işleme ve uygulama yeteneği.

İnsanların Yapay Zekâ'nın anlamını nasıl anladığı, mevcut algılarına bağlı olarak değişiklik göstermektedir. 1970'lerde, bir bilgisayar sisteminin bir insanı satrançta yenebileceği fikri gelecekte bir yerlerdeydi ve çoğu bu durumu Yapay Zekâ olarak kabul ediyordu. Şimdi, bilgisayar tabanlı sistem Deep Blue'nun dünya satranç şampiyonu Garry Kasparov'u yenmesinden yirmi yıl sonra, o sistemde uygulanan "kaba kuvvet" yaklaşımı, birçok kişi tarafından gerçek Yapay Zekâ olarak kabul edilmemektedir (yani, sistem verilerden öğrenme yeteneğine ve kendiliğinden öğrenme kapasitesine sahip değildir). Benzer şekilde, 1970'lerin ve 1980'lerin uzman sistemleri, uzmanın varlığı olmadan tekrar tekrar çalıştırılacak kurallar olarak insan uzmanlığını içermektedir. Bunlar o zamanlar Yapay Zekâ olarak kabul ediliyordu, ancak şimdi öyle kabul edilmemektedir.

Yapay Zekâ'nın algılanışındaki değişim "Yapay Zekâ (YZ) Etkisi" [R01] olarak bilinir. Toplumda Yapay Zekâ'nın algılanışı değiştiğinde, tanımı da değişir. Sonuç olarak, bugün yapılan herhangi bir tanım, gelecekte değişebilir ve geçmişten gelen tanımlarla uyumlayabilir.

1.2 Dar, Genel ve Süper YZ

Yüksek seviyede, Yapay Zekâ üç kategoriye ayrılabilir:

- Dar Yapay Zekâ (aynı zamanda zayıf Yapay Zekâ olarak da bilinir) sistemleri, sınırlı bağlamda belirli bir görevi yerine getirmek üzere programlanmıştır. Şu anda bu Yapay Zekâ formu yaygın olarak mevcuttur. Örneğin, oyun oynama sistemleri, spam filtreleri, test senaryosu üreticileri ve sesli asistanlar.
- Genel Yapay Zekâ (aynı zamanda güçlü Yapay Zekâ olarak da bilinir) sistemleri, insanlara benzer genel (geniş kapsamlı) bilişsel yeteneklere sahiptir. Bu Yapay Zekâ tabanlı sistemler, insanlar gibi kendi çevrelerini anlayabilir, akıl yürütebilir ve buna göre hareket edebilir. 2021 itibarıyla, herhangi bir genel Yapay Zekâ sistemi gerçekleştirilmemiştir.
- Süper Yapay Zekâ sistemleri, insan bilişini (genel Yapay Zekâ) kopyalama kapasitesine sahiptir ve muazzam işlem gücünden, neredeyse sınırsız hafızadan ve tüm insan bilgisine erişimden (örneğin internete erişim yoluyla) faydalanır. Süper Yapay Zekâ sistemlerinin, insanlardan hızlı bir şekilde daha bilgi hale geleceği düşünülmektedir. Yapay Zekâ tabanlı sistemlerin genel Yapay Zekâ'dan süper Yapay Zekâ'ya geçiş yaptığı nokta, genellikle teknolojik tekillik [B01] olarak bilinir.

1.3 YZ Tabanlı ve Geleneksel Sistemler

Geleneksel bir bilgisayar sistemi, tipik olarak insanlar tarafından if-then-else ve loops (döngüler) gibi yapıları içeren imperatif (yordamsal) bir dil kullanılarak programlanır. İnsanlar için sistemin girdileri çıktılarına nasıl dönüştürdüğünü anlaması nispeten kolaydır. Makine Öğrenimi (MÖ) kullanan Yapay Zekâ tabanlı bir sistemde ise, sistemin gelecekte yeni verilere nasıl tepki vereceğini belirlemek için verilerdeki örüntüler kullanılır. (Makine Öğrenimi'nin detaylı açıklaması için Bölüm 3'e bakınız). Örneğin, kedilerin görüntülerini tanımlamak için tasarlanmış bir yapay zekâ tabanlı görüntü işleyici, içinde kedilerin bulunduğu bilinen bir görüntü seti ile eğitilir. Yapay Zekâ, kendi başına verilerde kedileri tanımlamak için kullanılacak desenleri veya özellikleri belirler.

Bu desenler ve kurallar daha sonra yeni görüntülerde kedilerin olup olmadığını belirlemek için uygulanır.

Birçok Yapay Zekâ tabanlı sistemde, bu, tahmin yapma prosedürünün insanlar tarafından anlaşılmasını daha zor hale getirir (Bölüm 2.7'ye bakınız).

Uygulamada, Yapay Zekâ tabanlı sistemler çeşitli teknolojiler kullanılarak gerçekleştirilebilir (Bölüm 1.4'e bakınız) ve "Yapay Zekâ Etkisi" (Bölüm 1.1'e bakınız), şu anda neyin Yapay Zekâ tabanlı bir sistem olarak kabul edildiğini ve neyin geleneksel bir sistem olarak kabul edildiğini belirleyebilir.

1.4 YZ Teknolojileri

Yapay Zekâ, aşağıdakiler gibi çeşitli teknolojilerden faydalanılarak hayata geçirilebilir (daha fazla detay için [B02]'ye bakınız):

- Bulanık mantık
- Arama algoritmaları
- Akıl yürütme teknikleri
 - Kural motorları
 - Tümdengelimli sınıflandırıcılar
 - Vakaya dayalı akıl yürütme
 - Prosedürel akıl yürütme
- Makine Öğrenimi teknikleri
 - Sinir ağları
 - Bayes modelleri
 - Karar ağaçları
 - Rastgele orman
 - Doğrusal regresyon
 - Lojistik regresyon
 - Kümeleme algoritmaları
 - Genetik algoritmalar
 - Destek vektör makinesi (SVM)

Yapay Zekâ tabanlı sistemler genellikle bu teknolojilerden birini veya birkaçını uygular.

1.5 YZ Geliştirme Çerçeveleri

Belirli alanlara odaklanmış birçok Yapay Zekâ geliştirme çerçevesi mevcuttur. Bu çerçeveler, veri hazırlama, algoritma seçimi ve modellerin çeşitli işlemcilerde çalışacak şekilde derlenmesi gibi bir dizi faaliyeti destekler, örneğin merkezi işlem birimleri (MIB'ler), grafik işlem birimleri (GIB'ler) veya Bulut Tensor İşlem Birimleri (BTIB'ler). Belirli bir çerçevenin seçimi, kullanılan programlama dili ve kullanım kolaylığı gibi belirli hususlara da bağlı olabilir. Aşağıdaki çerçeveler en popüler olanlarından bazılarıdır (Nisan 2021 itibarıyla):

- Apache MxNet: Amazon'un Amazon Web Hizmetleri (AWS) için kullandığı açık kaynaklı bir derin öğrenme çerçevesi [R02].

- CNTK: Microsoft Cognitive Toolkit (CNTK), açık kaynaklı bir derin öğrenme araç setidir [R03].
- IBM Watson Studio: YZ çözümlerinin geliştirilmesini destekleyen bir dizi araç [R04].
- Keras: Python dilinde yazılmış, TensorFlow ve CNTK üzerinde çalışabilen yüksek seviyeli açık kaynaklı bir API [R06].
- PyTorch: Facebook tarafından işletilen ve uygulamalarında görüntü işleme ve doğal dil işleme (NLP) uygulayan açık kaynaklı bir Makine Öğrenimi kütüphanesi. Hem Python hem de C++ arayüzleri için destek sağlanmaktadır [R07].
- Scikit-learn: Python programlama dili için açık kaynaklı bir Makine Öğrenimi kütüphanesi [R08].
- TensorFlow: Google tarafından sağlanan, ölçeklenebilir Makine Öğrenimi için veri akışı grafiklerine dayalı açık kaynaklı bir MÖ çerçevesi [R05].

Bu geliştirme çerçeveleri sürekli olarak gelişmekte, bazen birleşmekte ve bazen yeni çerçevelerle değiştirilmektedir.

1.6 YZ Tabanlı Sistemler için Donanım

Makine Öğrenimi model eğitimi (Bölüm 3'e bakınız) ve model uygulaması için çeşitli donanımlar kullanılmaktadır. Örneğin, konuşma tanıma gerçekleştiren bir model, düşük seviyeli bir akıllı telefonda çalışabilir, ancak eğitilmesi için bulut bilişimin gücüne erişim gerekebilir. Ev sahibi cihaz internete bağlı değilse yaygın olarak kullanılan bir yaklaşım, modeli bulutta eğitmek ve sonra onu ev sahibi cihaza dağıtmaktır.

Makine Öğrenimi, genellikle aşağıdaki özellikleri destekleyen donanımdan yararlanır:

- Düşük hassasiyetli aritmetik: Bu, hesaplama için daha az bit kullanır (örneğin, Makine Öğrenimi için genellikle gereken 32-bit yerine 8 bit).
- Büyük veri yapılarıyla çalışabilme yeteneği (örneğin, matris çarpımını desteklemek için).
- Yoğun paralel (eş zamanlı) işleme.

Genel amaçlı CPU'lar, Makine Öğrenimi uygulamaları için genellikle gerekli olmayan karmaşık operasyonlar için destek sağlar ve sadece birkaç çekirdek sunar. Sonuç olarak, mimarileri, binlerce çekirdeğe sahip olan ve görüntülerin yoğun paralel ancak nispeten basit işlenmesini gerçekleştirmek üzere tasarlanmış GPU'lara kıyasla Makine Öğrenimi modellerini eğitme ve çalıştırma açısından daha az verimlidir. Bu nedenle, GPU'lar genellikle daha yüksek saat hızlarına sahip olsalar bile Makine Öğrenimi uygulamaları için CPU'lardan daha iyi performans gösterir. Küçük ölçekli Makine Öğrenimi çalışmaları için GPU'lar genellikle en iyi seçeneği sunar.

Bazı donanımlar özellikle Yapay Zekâ için tasarlanmıştır, örneğin özel olarak yapılmış Uygulamaya Özgü Entegre Devreler (ASIC'ler) ve Sistem Çipi (SoC). Bu Yapay Zekâ'ya özgü çözümler, çoklu çekirdekler, özel veri yönetimi ve bellek içi işleme yapabilme yeteneği gibi özelliklere sahiptir. Bunlar, Makine Öğrenimi modelinin eğitiminin bulutta yapıldığı kenar hesaplama için en uygun olanlardır.

Özel Yapay Zekâ mimarilerine sahip donanımlar şu anda (2021 Nisan itibarıyla) geliştirilmekte olan bir alandır. Bu, geleneksel von Neumann mimarisini kullanmayan, ancak daha ziyade beyin nöronlarını gevşek bir şekilde taklit eden bir mimariye sahip nöromorfik işlemcileri [B03] içerir.

Yapay Zekâ donanım sağlayıcıları ve işlemcilerine örnekler (2021 Nisan itibarıyla) şunlardır:

- NVIDIA: Bir dizi GPU ve Yapay Zekâ'ya özgü işlemciler sunar, örneğin Volta [R09].
- Google: Hem eğitim hem de çıkarsama için uygulamaya özgü entegre devreler geliştirmişlerdir. Google TPUs (Cloud Tensor İşlem Birimleri) [R10] kullanıcılar tarafından Google Cloud üzerinden erişilebilirken, Edge TPU [R11] bireysel cihazlarda Yapay Zekâ çalıştırmak için özel olarak yapılmış bir ASIC'tir.

- Intel: Derin öğrenme (hem eğitim hem de çıkarsama) için Nervana sinir ağı işlemcilerini ve bilgisayar görüşü ile sinir ağı uygulamalarında çıkarsama için Movidius Myriad görüş işleme birimlerini sunar.
- Mobileye: Karmaşık ve hesaplama yoğunluğu yüksek görüntü işleme desteği için EyeQ ailesi SoC cihazları üretirler [R13]. Bunlar, araçlarda kullanım için düşük güç tüketimine sahiptir.
- Apple: iPhone'larda cihaz içi Yapay Zekâ için Bionic çip üretirler [B04].
- Huawei: Akıllı telefonlar için Kirin 970 çipi, Yapay Zekâ için yerleşik sinir ağı işleme özelliğine sahiptir [B05].

1.7 YZ Servisleri (AlaaS)

Yapay Zekâ bileşenleri, örneğin Makine Öğrenimi modelleri, bir organizasyon içinde oluşturulabilir, üçüncü bir partiden indirilebilir veya web üzerinde bir hizmet olarak kullanılabilir (AlaaS). Bazı Yapay Zekâ işlevselliğinin sistem içinden sağlandığı ve bazısının bir hizmet olarak sunulduğu hibrit bir yaklaşımda mümkündür.

Makine Öğrenimi bir hizmet olarak kullanıldığında, web üzerinden bir Makine Öğrenimi modeline erişim sağlanır ve veri hazırlama, depolama, model eğitimi, değerlendirme, ayarlama, test etme ve dağıtım için destek de sunulabilir.

Üçüncü taraf sağlayıcılar (örn., AWS, Microsoft) yüz ve konuşma tanıma gibi belirli Yapay Zekâ hizmetleri sunar. Bu, kendi Yapay Zekâ hizmetlerini oluşturmak için yeterli kaynak ve uzmanlığa sahip olmayan bireylerin ve organizasyonların bulut tabanlı hizmetler kullanarak Yapay Zekâ uygulamasına olanak tanır. Ayrıca, üçüncü taraf bir hizmetin parçası olarak sunulan Makine Öğrenimi modelleri, Yapay Zekâ piyasasına yeni girenler gibi birçok paydaşın kolayca erişebileceğinden daha büyük ve daha çeşitli bir eğitim veri seti üzerinde eğitilmiş olma olasılığı daha yüksektir.

1.7.1 YZ Servis Sözleşmeleri

Bu Yapay Zekâ hizmetleri genellikle, Yapay Zekâ olmayan bulut tabanlı Hizmet olarak Yazılım (SaaS) ile benzer sözleşmelerle sağlanır. AlaaS sözleşmesi genellikle erişilebilirliği ve güvenlik taahhütlerini tanımlayan bir hizmet düzeyi sözleşmesi (SLA) içerir. Bu tür SLA'lar genellikle hizmetin çalışma süresini (örn., %99,99 çalışma süresi) ve hataları düzeltmek için bir yanıt süresini kapsar ancak nadiren Makine Öğrenimi işlevsel performans metriklerini (örneğin, doğruluk) benzer bir şekilde tanımlar (Bölüm 5'e bakınız). AlaaS genellikle abonelik esasında ödenir ve sözleşmede belirtilen erişilebilirlik ve/veya yanıt süreleri karşılanmazsa, hizmet sağlayıcı genellikle gelecekteki hizmetler için krediler sağlar. Bu krediler dışında, çoğu AlaaS sözleşmesi, ödenen ücretler dışında sınırlı bir sorumluluk sağlar, bu da AlaaS'a bağlı olan YZ tabanlı sistemlerin genellikle nispeten düşük riskli uygulamalarla sınırlı olduğu anlamına gelir, burada hizmet kaybı çok zararlı olmaz.

Hizmetler genellikle bir kabul dönemi yerine başlangıçta ücretsiz deneme süresi ile gelir. Bu dönemde, AlaaS'ın tüketicisinin sağlanan hizmetin ihtiyaç duyulan işlevsellik ve performans açısından (örn., doğruluk) ihtiyaçlarını karşılayıp karşılamadığını test etmesi beklenir. Bu, genellikle sağlanan hizmetin şeffaflığındaki herhangi bir eksikliği kapsamak için gereklidir (Bölüm 7.5'e bakınız).

1.7.2 Yapay Zekâ Servis Örnekleri (AlaaS)

Aşağıdakiler, Nisan 2021 itibarıyla AlaaS örnekleridir:

- IBM Watson Assistant: Bu, aylık aktif kullanıcı sayısına göre fiyatlandırılan bir Yapay Zekâ sohbet robotudur.
- Google Cloud Yapay Zekâ ve Makine Öğrenimi Ürünleri: Bunlar, form analizörü ve belge OCR'ini içeren belge tabanlı Yapay Zekâ sağlar. Fiyatlar, işlenmek üzere gönderilen sayfa sayısına göre belirlenir.

- Amazon CodeGuru: Geliştiricilere kod kalitelerini iyileştirme konusunda öneriler sunan Makine Öğrenimi Java kodunun bir incelemesini sağlar. Fiyatlar, analiz edilen kaynak kodunun satır sayısına göre belirlenir.
- Microsoft Azure Cognitive Search: Yapay Zekâ bulut araması sağlar. Fiyatlar, arama birimlerine (kullanılan depolama ve veri aktarımı açısından tanımlanır) göre belirlenir.

1.8 Eğitim Öncesi Modeller

1.8.1 Eğitim Öncesi Modellere Giriş

Makine Öğrenimi modellerini eğitmek pahalı olabilir (Bölüm 3'e bakınız). Öncelikle, veriler hazırlanmalı ve sonra model eğitilmelidir. İlk aktivite büyük miktarda insan kaynağı tüketebilirken, sonraki aktivite çok fazla hesaplama kaynağı tüketebilir. Birçok organizasyonun bu kaynaklara erişimi bulunmamaktadır.

Daha ucuz ve çoğu zaman daha etkili bir alternatif, önceden eğitilmiş bir model kullanmaktır. Bu, gereken modele benzer işlevsellik sağlar ve önceden eğitilmiş modelin işlevselliğini genişleten ve/veya odaklayan yeni bir model oluşturmak için bir temel olarak kullanılır. Bu tür modeller, sinir ağları ve rastgele ormanlar gibi sınırlı sayıda teknoloji için mevcuttur.

Bir görüntü sınıflandırıcısına ihtiyaç duyulursa, 14 milyondan fazla görüntünün 1000'den fazla kategoriye sınıflandırıldığı kamuoyuna açık ImageNet veri seti kullanılarak eğitilebilir. Bu, başarı garantisi olmadan önemli kaynaklar tüketme riskini azaltır. Alternatif olarak, bu veri seti üzerinde zaten eğitilmiş olan mevcut bir model yeniden kullanılabilir. Böyle bir önceden eğitilmiş model kullanılarak, eğitim maliyetlerinden tasarruf edilir ve işe yaramama riski büyük ölçüde ortadan kaldırılır.

Önceden eğitilmiş bir model, değişiklik yapılmadan kullanıldığında, Yapay Zekâ tabanlı sistemde basitçe gömülebilir veya bir hizmet olarak kullanılabilir (1.7 Bölümüne bakınız).

1.8.2 Transfer Öğrenme

Önceden eğitilmiş bir model alınıp, ikinci, farklı bir gereksinimi yerine getirecek şekilde değiştirilmesi de mümkündür. Bu, transfer öğrenme olarak bilinir ve derin sinir ağlarında kullanılır; burada, sinir ağının erken katmanları (Bölüm 6'ya bakınız) genellikle oldukça temel görevler gerçekleştirir (örneğin, bir görüntü sınıflandırıcısında düz ve eğri çizgiler arasındaki farkı belirlemek), oysa sonraki katmanlar daha uzmanlaşmış görevler gerçekleştirir (örneğin, bina mimarisi türleri arasında ayırım yapmak). Bu örnekte, bir görüntü sınıflandırıcısının son katmanları dışındakiler yeniden kullanılabilir, erken katmanları eğitime ihtiyacını ortadan kaldırır. Sonra, son katmanlar yeni bir sınıflandırıcı için benzersiz gereksinimleri ele alacak şekilde yeniden eğitilir. Pratikte, önceden eğitilmiş model, probleme özgü yeni verilerle ek eğitim yapılarak ince ayar yapılabilir.

Bu yaklaşımın etkililiği, büyük ölçüde orijinal modelin gerçekleştirdiği işlev ile yeni modelin gerektirdiği işlev arasındaki benzerliğe bağlıdır. Örneğin, kedi türlerini tanımlayan bir görüntü sınıflandırıcısını değiştirip sonra köpek ırklarını tanımlamasını sağlamak, onu insanların aksanlarını tanımlayacak şekilde değiştirmekten çok daha etkili olur.

Özellikle akademik araştırmacıların hazırladığı birçok önceden eğitilmiş model mevcuttur. Bu tür önceden eğitilmiş modellere örnekler arasında, görüntü sınıflandırma için ImageNet modelleri [R14] (Inception, VGG, AlexNet ve MobileNet) ve Google'ın BERT [R15] gibi önceden eğitilmiş NLP modelleri bulunmaktadır.

1.8.3 Eğitim Öncesi Modelleri ve Transfer Öğrenmeyi Kullanmanın Riskleri

Önceden eğitilmiş modelleri ve transfer öğrenmeyi kullanmak, Yapay Zekâ tabanlı sistemler inşa etmek için yaygın yaklaşımlar olsa da bunlarla ilişkilendirilen bazı riskler bulunmaktadır. Bunlar arasında:

- Önceden eğitilmiş bir model, içsel olarak üretilen bir modele kıyasla şeffaflık eksikliği gösterebilir.
- Önceden eğitilmiş modelin gerçekleştirdiği işlev ile gerekli işlevsellik arasındaki benzerlik seviyesi yetersiz olabilir. Ayrıca, bu fark veri bilimcisi tarafından anlaşılabilir.
- Önceden eğitilmiş modelin orijinal olarak geliştirildiğinde kullanılan veri hazırlama adımları (Bölüm 4.1'e bakınız) ile bu model daha sonra yeni bir sistemde kullanıldığında kullanılan veri hazırlama adımları arasındaki farklar, sonuçlanan işlevsel performansı etkileyebilir.
- Önceden eğitilmiş bir modelin eksiklikleri, onu yeniden kullananlar tarafından miras alınabilir ve bu eksiklikler belgelenmiş olmayabilir. Örneğin, miras alınan yanlılıklar (Bölüm 2.4'e bakınız) eğitilmiş model için kullanılan veriler hakkında yeterli belge yoksa açık olmayabilir. Ayrıca, önceden eğitilmiş model geniş çapta kullanılmıyorsa, daha fazla bilinmeyen (veya belgelenmemiş) hata olabilir ve bu riski hafifletmek için daha titiz testler gerekebilir.
- Transfer öğrenme yoluyla oluşturulan modeller, üzerine kurulduğu önceden eğitilmiş modelle aynı zayıflıklara karşı oldukça hassas olma ihtimali yüksektir (örneğin, 9.1.1'de açıklandığı gibi yanıltıcı saldırılar). Ek olarak, bir Yapay Zekâ tabanlı sistemin belirli bir önceden eğitilmiş modeli içerdiği (veya belirli bir önceden eğitilmiş modele dayandığı) biliniyorsa, potansiyel saldırganlar tarafından zaten bilinen zayıflıklarla ilişkilendirilebilir.

Yukarıdaki risklerin birçoğu, önceden eğitilmiş model için kapsamlı belgelendirme mevcut olduğunda daha kolay hafifletilebilir (Bölüm 7.5'e bakınız).

1.9 Standartlar, Yönetmelikler ve YZ

IEC ve ISO'nun bilgi teknolojisi üzerine Ortak Teknik Komitesi (ISO/IEC JTC1), Yapay Zekâ'ya katkıda bulunan uluslararası standartlar hazırlar. Örneğin, bir Yapay Zekâ alt komitesi (ISO/IEC JTC 1/SC42) 2017'de kurulmuştur. Ayrıca, yazılım ve sistem mühendisliğini kapsayan ISO/IEC JTC1/SC7, "Yapay Zekâ Tabanlı Sistemlerin Test Edilmesi" üzerine bir teknik rapor yayımlamıştır [S01]. Yapay Zekâ üzerine standartlar ayrıca bölgesel düzeyde (örneğin, Avrupa standartları) ve ulusal düzeyde de yayımlanmaktadır.

AB genelinde Genel Veri Koruma Yönetmeliği (GDPR) Mayıs 2018'de yürürlüğe girmiş ve kişisel veriler ile otomatik karar alma konusunda veri işleyicilerine yönelik yükümlülükler belirlemiştir [B06]. Potansiyel ayrımcılığın azaltılmasını da içeren Yapay Zekâ sistem işlevsel performansının değerlendirilmesi ve iyileştirilmesi gerekliliklerini, ayrıca bireylerin otomatik karar almaya tabi tutulmama haklarının sağlanmasını içerir. GDPR'nin test açısından en önemli yönü, kişisel verilerin (tahminler dahil) doğru olması gerektiğidir. Bu, sistemin yaptığı her bir tahminin mutlaka doğru olması gerektiği anlamına gelmez, ancak sistem, kullanıldığı amaçlar için yeterince doğru olmalıdır.

Alman ulusal standartlar kurumu (DIN) da YZ Kalite Metamodeli'ni geliştirmiştir ([S02], [S03]).

Yapay Zekâ üzerine standartlar ayrıca endüstri kuruluşları tarafından da yayımlanmaktadır. Örneğin, Elektrik ve Elektronik Mühendisleri Enstitüsü (IEEE), etik ve Yapay Zekâ üzerine bir dizi standart üzerinde çalışmaktadır (IEEE Küresel Etik Düşünceler Yapay Zekâ ve Otonom Sistemler Girişimi). Bu standartların birçoğu yazıldığı sırada hala geliştirilme aşamasındadır.

Yapay Zekâ, güvenlikle ilgili sistemlerde kullanıldığında, ilgili düzenleyici standartlar geçerlidir, örneğin otomotiv sistemleri için ISO 26262 [S04] ve ISO/PAS 21448 (SOTIF) [S05]. Bu tür düzenleyici standartlar tipik olarak devlet organları tarafından zorunlu kılınır ve bazı ülkelerde dahil edilen yazılım ISO 26262'ye uyumlu olmadan bir aracın satılması yasa dışı olabilir. Standartlar izole edilmiş belgelerdir ve kullanımları normalde yalnızca yasalar veya sözleşmeler tarafından zorunlu kılınır. Ancak, birçok standart kullanıcıyı, yazarların uzmanlığından faydalanmak ve daha yüksek kaliteli ürünler oluşturmak için standartları kullanır.

2.Yapay Zekâ Tabanlı Sistemler için Kalite Özellikleri – 105 Dakika

Anahtar Kelimeler

Yok

YZ-Spesifik Anahtar Kelimeleri

Uyarlanabilirlik, Algoritmik Yanlılık, Otonomi, Yanlılık, Evrim, Açıklanabilirlik, Açıklanabilir Yapay Zekâ (XAI), Esneklik, Uygunsuz Yanlılık, Yorumlanabilirlik, MÖ (Makine Öğrenimi) Sistemi, Makine Öğrenimi, Ödül Hileciliği, Sağlık, Örneklem Yanlılık, Kendi Kendine Öğrenme Sistemi, Yan Etkiler, Şeffaflık

2. Bölüm Öğrenim Hedefleri:

2.1 Esneklik ve Uyarlanabilirlik

YZ-2.1.1 K2 Yapay Zekâ tabanlı sistemlerin özellikleri olarak esneklik ve uyarlanabilirliğin önemini açıklayın.

2.2 Otonomi

YZ-2.2.1 K2 **Otonomi** ile Yapay Zekâ tabanlı sistemler arasındaki ilişkiyi açıklayın.

2.3 Evrim

YZ-2.3.1 K2 Yapay Zekâ tabanlı sistemler için evrimi yönetmenin önemini açıklayın.

2.4 Yanlılık

YZ-2.4.1 K2 Yapay Zekâ tabanlı sistemlerde bulunan farklı nedenleri ve yanlılık türlerini açıklayın.

2.5 Etik

YZ-2.5.1 Yapay Zekâ tabanlı sistemlerin geliştirilmesi, dağıtımı ve kullanımında saygı duyulması gereken etik ilkeleri tartışın.

2.6 Yan Etkiler ve Ödül Hileciliği

YZ-2.6.1 K2 Yapay Zekâ tabanlı sistemlerde yan etkilerin ve ödül hileciliğinin ortaya çıkışını açıklayın.

2.7 Şeffaflık, Yorumlanabilirlik ve Açıklanabilirlik

YZ-2.7. Yapay Zekâ tabanlı sistemlerde şeffaflığın, yorumlanabilirliğin ve açıklanabilirliğin nasıl geçerli olduğunu açıklayın.

2.8 Güvenlik ve Yapay Zekâ

YZ-2.8.1 K1 Yapay Zekâ tabanlı sistemlerin güvenlikle ilgili uygulamalarda kullanılmasını zorlaştıran özellikleri hatırlayın.

2.1 Esneklik ve Uyarlanabilirlik

Esneklik ve uyarlanabilirlik, yakından ilişkili kalite özellikleridir. Bu müfredatta esneklik, sistemin orijinal gereksinimlerinin bir parçası olmayan durumlarda kullanılabilme yeteneği olarak kabul edilirken; uyarlanabilirlik, sistemin farklı donanımlar ve değişen operasyonel ortamlar gibi yeni durumlara kolayca uyum sağlayabilme yeteneği olarak kabul edilmektedir.

Hem esneklik hem de uyarlanabilirlik aşağıdaki durumlarda faydalıdır:

- sistem devreye alındığında operasyonel ortam tam olarak bilinmiyorsa
- sistemin yeni operasyonel ortamlarla başa çıkması bekleniyorsa
- sistemin yeni durumlara uyum sağlaması bekleniyorsa
- sistemin davranışını ne zaman değiştireceğini belirlemesi gerekiyorsa.

Kendi kendine öğrenen Yapay Zekâ tabanlı sistemlerin yukarıdaki özelliklerin tümünü göstermesi bekleniyor. Sonuç olarak uyarlanabilir olmaları ve esnek olma potansiyeline sahip olmaları gerekir.

Bir Yapay Zekâ tabanlı sistemin esneklik ve uyarlanabilirlik gereksinimleri, sistemin uyum sağlaması beklenen herhangi bir çevre değişikliğinin detaylarını içermelidir. Bu gereksinimler ayrıca sistemin kendini uyarlamak için kullanabileceği zaman ve kaynaklar üzerindeki kısıtlamaları da belirtmelidir (örneğin, yeni bir nesne türünü tanımak için ne kadar süre alabilir).

2.2 Otonomi

Otonomi tanımlanırken, öncelikle otonom bir sistemin insan gözetimi ve kontrolünden tamamen bağımsız olacağını tanımak önemlidir. Pratikte, tam otonomi genellikle arzu edilmez. Örneğin, genellikle "otonom" olarak adlandırılan tamamen kendi kendine gidebilen araçlar resmi olarak "tam sürüş otomasyonuna" sahip olarak sınıflandırılmaktadır [B07].

Birçok kişi, otonom sistemleri "akıllı" veya "zeki" olarak değerlendirir, bu da belirli işlevleri yerine getirmek için Yapay Zekâ tabanlı bileşenler içereceklerini önerir. Örneğin, durumsal olarak farkında olması gereken otonom araçlar tipik olarak, aracın hemen çevresi hakkında bilgi toplamak için birkaç sensör ve görüntü işleme kullanır. Makine Öğrenimi ve özellikle derin öğrenme (Bkz. Bölüm 6.1), bu işlevi yerine getirmede en etkili yaklaşım olarak bulunmuştur. Otonom sistemler ayrıca karar alma ve kontrol işlevlerini de içerebilir. Her ikisi de Yapay Zekâ tabanlı bileşenler kullanılarak etkili bir şekilde gerçekleştirilebilir.

Bazı Yapay Zekâ tabanlı sistemler otonom olarak kabul edilse de bu her Yapay Zekâ tabanlı sisteme uygulanmaz. Bu ders programında, otonomi, sistemin uzun süreler boyunca insan gözetimi ve kontrolünden bağımsız olarak çalışabilme yeteneği olarak kabul edilir. Bu, belirli ve test edilmesi gereken bir otonom sistemin özelliklerini belirlemeye yardımcı olabilir. Örneğin, otonom bir sistemin insan müdahalesi olmadan tatmin edici bir şekilde ne kadar performans gösterebileceğinin bilinmesi gereklidir. Ayrıca, otonom sistemin kontrolü insan kontrolörlerine geri vermesi gereken olayların belirlenmesi önemlidir.

2.3 Evrim

Bu ders programında, evrim, sistemin dışsal kısıtlamalardaki değişikliklere yanıt olarak kendini iyileştirme yeteneği olarak kabul edilir. Bazı Yapay Zekâ sistemleri kendi kendine öğrenen olarak tanımlanabilir ve başarılı kendi kendine öğrenen Yapay Zekâ tabanlı sistemler, bu tür bir evrimi içermelidir.

Yapay Zekâ tabanlı sistemler genellikle gelişen bir çevrede işlerler. Diğer bilgi teknolojileri sistemleri gibi, bir Yapay Zekâ tabanlı sistemin işletim çevresindeki değişikliklerle başa çıkacak kadar esnek ve uyarlanabilir olması gereklidir. Kendi kendine öğrenen Yapay Zekâ tabanlı sistemler tipik olarak iki tür değişikliği yönetmelidir:

- Değişimin bir biçimi, sistemin kendi kararlarından ve çevresiyle olan etkileşimlerinden öğrenmesidir.
- Diğer değişim biçimi ise, sistemin işletim çevresinde yapılan değişikliklerden öğrenmesidir.

Her iki durumda da sistem ideal olarak etkinliğini ve verimliliğini artırmak için evrimleşmelidir. Ancak, bu evrim, sistemin istenmeyen özellikler geliştirmesini önlemek için sınırlandırılmalıdır. Herhangi bir evrim, orijinal sistem gereksinimleri ve kısıtlamalarıyla uyumlu olmaya devam etmelidir. Bunlar eksik olduğunda, sistem, herhangi bir evrimin sınırlar içinde kalmasını ve her zaman insan değerleriyle uyumlu kalmasını sağlamak üzere yönetilmelidir. Bölüm 2.6, kendi kendine öğrenen Yapay Zekâ tabanlı sistemlerde yan etkilerin ve ödül hilesinin etkisine ilişkin örnekler sunmaktadır.

2.4 Yanlılık

Yapay Zekâ tabanlı sistemler bağlamında yanlılık, sistemin sağladığı çıktılar ile 'tarafsız çıktılar' olarak kabul edilen ve belirli bir gruba ayrıcalık göstermeyen çıktılar arasındaki mesafenin istatistiksel bir ölçüsüdür. Uygunsuz yanlılıklar, cinsiyet, ırk, etnik köken, cinsel yönelim, gelir seviyesi ve yaş gibi niteliklerle bağlantılı olabilir. Yapay Zekâ tabanlı sistemlerde, örneğin banka kredisi önerileri, işe alım sistemleri ve yargı izleme sistemlerinde, uygunsuz yanlılık vakaları bildirilmiştir.

Yanlılık, birçok türde Yapay Zekâ tabanlı sisteme sokulabilir. Örneğin, uzmanların yanlılığının bir uzman sistem tarafından uygulanan kurallara dahil edilmesini önlemek zordur. Ancak, Makine Öğrenimi sistemlerinin yaygınlığı, yanlılık ile ilgili tartışmaların genellikle bu sistemlerin bağlamında gerçekleştiği anlamına gelir. Makine Öğrenimi sistemleri, toplanan verileri kullanarak kararlar ve tahminler yapmak için algoritmaları kullanır ve bu iki bileşen sonuçlarda yanlılığa neden olabilir:

- Algoritmik yanlılık, öğrenme algoritması yanlış yapılandırıldığında ortaya çıkabilir, örneğin, bazı verilerin diğerlerine göre fazla değerlendirildiği durumlar. Bu yanlılık kaynağı, Makine Öğrenimi algoritmalarının hiperparametre ayarlamasıyla yönetilebilir ve oluşturulabilir (Bkz. Bölüm 3.2).
- Örneklem yanlılığı, eğitim verileri MÖ'nün uygulandığı veri alanını tam olarak temsil etmediğinde ortaya çıkabilir.

Uygunsuz yanlılık genellikle örneklem yanlılığı nedeniyle oluşur, ancak ara sıra algoritmik yanlılık tarafından da neden olabilir.

2.5 Etik

Etik, Cambridge Sözlüğü'nde şu şekilde tanımlanmaktadır:

Özellikle ahlaki temellere dayanan ve davranışları kontrol eden kabul görmüş inançlar bütünüdür.

Gelişmiş yeteneklere sahip Yapay Zekâ tabanlı sistemler, insanların hayatları üzerinde genellikle olumlu bir etkiye sahiptir. Bu sistemler daha yaygın hale geldikçe, etik bir şekilde kullanılıp kullanılmadıkları konusunda endişeler ortaya çıkmıştır.

Etik olarak kabul edilen şey zamanla ve ayrıca yerellikler ve kültürler arasında da değişebilmektedir.

Bir Yapay Zekâ tabanlı sistemin bir konumdan diğerine dağıtımı, paydaş değerlerindeki farklılıkları göz önünde bulundurulmalıdır.

Yapay Zekâ etiği konusunda ulusal ve uluslararası politikalar birçok ülke ve bölgede bulunabilir.

Ekonomik İş birliği ve Kalkınma Örgütü, hükümetler tarafından Yapay Zekâ'nın sorumlu geliştirilmesi için kabul edilen ilk uluslararası standartlar olan Yapay Zekâ için ilkelerini 2019'da yayınlamıştır [B08].

Bu ilkeler yayınlandıklarında kırk iki ülke tarafından benimsenmiştir ve ayrıca Avrupa Komisyonu tarafından da desteklenmektedir. Bunlar, "güvenilir Yapay Zekâ'nın sorumlu yönetimi" için pratik politika önerileri ile birlikte değere dayalı ilkeleri içermektedir. Bunlar özetle şöyledir:

- Yapay Zekâ, kapsayıcı büyümeyi, sürdürülebilir kalkınmayı ve refahı teşvik ederek insanlara ve gezegene fayda sağlamalıdır.
- Yapay Zekâ sistemleri, hukukun üstünlüğüne, insan haklarına, demokratik değerlere ve çeşitliliğe saygı göstermeli ve adil bir toplum sağlamak için uygun güvenlik önlemleri içermelidir.
- İnsanların sonuçları anlamasını ve bunlara itiraz edebilmesini sağlamak için YZ konusunda şeffaflık olmalıdır.
- Yapay Zekâ sistemleri, yaşam döngüleri boyunca sağlam, güvenli ve emniyetli bir şekilde işlev görmeli ve riskler sürekli olarak değerlendirilmelidir.
- Yapay Zekâ sistemlerini geliştiren, dağıtan veya işleten kuruluşlar ve bireyler sorumlu tutulmalıdır.

2.6. Yan Etkiler ve Ödül Hileciliği

Yapay Zekâ tabanlı sistemler, sistemin hedeflerine ulaşmaya çalışırken beklenmedik ve hatta zararlı sonuçlar üretebilir. Bu, yan etkiler ve ödül hileciliği ile sonuçlanabilir [B09].

Olumsuz yan etkiler, yapay zekâ tabanlı bir sistemin tasarımcısı, çevredeki belirli görevleri yerine getirmeye odaklanan ancak çevrenin diğer (potansiyel olarak çok geniş) unsurlarını göz ardı eden bir hedef belirlediğinde ortaya çıkabilir. Bu durum, aslında değiştirilmesi zararlı olabilecek çevresel değişkenlere karşı örtük bir kayıtsızlık ifade edilmesine yol açar. [B09]. Örneğin, "mümkün olduğunca yakıt tasarruflu ve güvenli bir şekilde" hedefine ulaşmak üzere tasarlanmış bir otonom araç, hedefine ulaşabilir, ancak yan etkisi yolcuların aşırı zaman almasından dolayı son derece rahatsız olmasıdır.

Ödül hileciliği, yapay zekâ tabanlı bir sistemin, belirli bir hedefe ulaşmak için "zekice" ya da "kolay" bir çözüm kullanarak "tasarımcının niyetinin özünü saptırması" sonucunda ortaya çıkabilir. Etkili bir şekilde, hedef manipüle edilebilir. Ödül hileciliğine yaygın bir örnek, Yapay Zekâ tabanlı bir sistemin kendini bir atari oyunu oynamayı öğretmesidir. Sistem, "en yüksek skoru elde etme" hedefiyle karşı karşıya kalır ve bunu yapmak için oyunu oynamak yerine en yüksek skoru saklayan veri kaydını ele geçirir.

2.7 Şeffaflık, Yorumlanabilirlik ve Açıklanabilirlik

Yapay Zekâ tabanlı sistemler genellikle, kullanıcıların bu sistemlere güvenmesi gereken alanlarda uygulanır. Bu, güvenlik nedenleriyle olabileceği gibi, gizliliğin gerektiği ve potansiyel olarak hayatı değiştirebilecek tahminlerde ve kararlarda bulunabilecekleri durumlar için de geçerlidir.

Çoğu kullanıcı, Yapay Zekâ tabanlı sistemleri "kara kutu" olarak görür ve bu sistemlerin sonuçlarına nasıl ulaştıklarından pek haberdar değildir. Bazı durumlarda, bu bilgisizlik, sistemleri inşa eden veri bilimcileri için de geçerli olabilir. Arada sırada, kullanıcılar Yapay Zekâ tabanlı bir sistemle etkileşimde olduklarının farkında bile olmayabilirler.

Yapay Zekâ tabanlı sistemlerin doğal karmaşıklığı, "Açıklanabilir Yapay Zekâ" (XAI) alanına yol açmıştır. XAI'nin amacı, kullanıcıların Yapay Zekâ tabanlı sistemlerin sonuçlarına nasıl ulaştığını anlayabilmelerini sağlamak, böylece kullanıcıların bu sistemlere olan güvenlerini artırmaktır.

Kraliyet Topluluğu'na göre [B10], Açıklanabilir Yapay Zekâ (XAI) istenmesinin birkaç nedeni bulunmaktadır, bunlar arasında:

- kullanıcılara sistemde güven sağlama
- yanlılığa karşı korunma
- düzenleyici standartlar veya politika gereksinimlerini karşılama
- sistem tasarımını iyileştirme
- risk, sağlamlık ve kırılabilirlik değerlendirme
- bir sistemden çıkan sonuçları anlama ve doğrulama
- otonom, yetkinlik (kullanıcıyı güçlendirilmiş hissettirme) ve sosyal değerleri karşılama

Bu, bir paydaşın perspektifinden Yapay Zekâ tabanlı sistemler için aşağıdaki üç temel, arzu edilen 'Açıklanabilir Yapay Zekâ (XAI)' özelliğine yol açar (ayrıca Bölüm 8.6'ya bakınız):

- Şeffaflık: Bu, modeli üretmek için kullanılan algoritma ve eğitim verilerinin ne kadar kolay belirlenebileceği olarak kabul edilir.
- Yorumlanabilirlik: Bu, kullanıcılar dahil olmak üzere çeşitli paydaşlar tarafından Yapay Zekâ teknolojisinin anlaşılabilirliği olarak kabul edilir.
- Açıklanabilirlik: Bu, kullanıcıların Yapay Zekâ tabanlı sistemin belirli bir sonuca nasıl ulaştığını ne kadar kolay belirleyebileceği olarak kabul edilir.

2.8 Güvenlik ve YZ

Bu ders programında, güvenlik, bir Yapay Zekâ tabanlı sistemin insanlara, mülke veya çevreye zarar vermemesi beklentisi olarak kabul edilir. Yapay Zekâ tabanlı sistemler, güvenliği etkileyebilecek kararlar almak için kullanılabilir. Örneğin; tıp, imalat, savunma, güvenlik ve ulaşım alanlarında çalışan Yapay Zekâ tabanlı sistemlerin güvenliği etkileme potansiyeli vardır.

Yapay Zekâ tabanlı sistemlerin güvenli olduklarını (örneğin, insanlara zarar vermemelerini) sağlamanın daha zor olmasına neden olan özellikler şunları içerir:

- karmaşıklık
- belirsizlik
- olasılıksal doğa
- kendi kendine öğrenme
- şeffaflık, yorumlanabilirlik ve açıklanabilirlik eksikliği
- sağlamlık eksikliği

Bu özelliklerin birkaçının test edilmesiyle ilgili zorluklar 8. Bölümde ele alınmıştır.

3. Makine Öğrenimi (MÖ) – Genel Bakış – 145 dakika

Anahtar Kelimeler

Yok

YZ-Spesifik Anahtar Kelimeleri

İlişkilendirme, Sınıflandırma, Kümeleme, Veri Hazırlama, Makine Öğrenme (MÖ) Algoritması, Makine Öğrenme (MÖ) Çerçevesi, Makine Öğrenme (MÖ) Fonksiyonel Performans Kriterleri, Makine Öğrenme (MÖ) Modeli, Makine Öğrenme (MÖ) Eğitim Verileri, Makine Öğrenme (MÖ) İş Akışı, Model Değerlendirmesi, Model Ayarlama, Aykırı Değer, Aşırı Uyum, Regresyon, Takviyeli Öğrenme, Denetimli Öğrenme, Yetersiz Uyum, Denetimsiz Öğrenme

Bölüm 3 için Öğrenme Hedefleri:

3.1 Makine Öğrenimi (MÖ) Türleri

YZ-3.1.1 K2 Denetimli öğrenmenin bir parçası olarak sınıflandırma ve regresyonu açıklayın.

YZ-3.1.2 K2 Denetimsiz öğrenmenin bir parçası olarak kümelemeyi ve ilişkilendirmeyi açıklayın.

YZ-3.1.3 K2 Takviyeli öğrenmeyi açıklayın.

3.2 Makine Öğrenimi (MÖ) İş Akışı

YZ-3.2.1 K2 Bir MÖ sistemini oluşturmak için kullanılan iş akışını özetleyin.

3.3 Bir Makine Öğrenimi (MÖ) Türü Seçme

YZ-3.3.1 K3 Bir proje senaryosu verildiğinde, uygun bir Makine Öğrenimi biçimi belirleyin (sınıflandırmadan, regresyon, kümeleme, ilişkilendirme veya takviyeli öğrenme).

3.4 Makine Öğrenimi (MÖ) Algoritması Seçiminde yer alan faktörler

YZ-3.4.1 K2 MÖ Algoritmalarının seçiminde rol oynayan faktörleri açıklayın.

3.5 Aşırı uyum ve yetersiz uyum

YZ-3.5.1 K2 Yetersiz uyum ve aşırı uyum kavramlarını özetleyin.

HO-3.5.1 H0 Yetersiz uyumu ve aşırı uyumu gösterin.

3.1 MÖ Çeşitleri

Makine Öğrenimi (MÖ) algoritmaları şu şekilde kategorize edilebilir:

- denetimli öğrenme (supervised learning),
- denetimsiz öğrenme (unsupervised learning) ve
- takviyeli öğrenme (reinforcement learning) .

3.1.1 Denetimli Öğrenme

Bu tür öğrenmede algoritma, eğitim sırasında etiketli verilerden MÖ modelini oluşturur. Tipik olarak girdi çiftlerinden oluşan etiketli veriler (örneğin, bir köpeğin görüntüsü ve "köpek" etiketi), algoritmanın girdi verileri (örneğin köpek resimleri) ile çıktı etiketleri (örneğin "köpek" ve "kedi") arasındaki ilişkiyi çıkarmak için kullanılır. MÖ modeli testi aşamasında, çıktıyı tahmin etmek için eğitilmiş modele yeni bir görünmeyen veri seti uygulanır. Model, çıktı doğruluk düzeyi tatmin edici olduğunda devreye alınır.

Denetimli öğrenmeyle çözülen problemler iki kategoriye ayrılır:

- **Sınıflandırma:** Bu, bir girdinin önceden tanımlanmış birkaç sınıftan birine atanması gerektirdiği problemlerde kullanılır. Bir görüntüde yüz tanıma veya nesne algılama, sınıflandırmayı kullanan problemlere örnektir.
- **Regresyon:** Bu, bir problem MÖ modelinin sayısal bir çıktı tahmin etmesini gerektirdiğinde kullanılır. Bir kişinin alışkanlıkları hakkında girdi verilerine dayanarak yaşını tahmin etmek veya gelecekteki hisse senedi fiyatlarını tahmin etmek gibi problemler, regresyon kullanılarak çözülen problemlere örnektir.

Dikkat edilmesi gereken nokta, bir MÖ problemi bağlamında kullanılan regresyon teriminin, diğer ISTQB® müfredatlarında olduğu gibi [I01], yazılım değişikliklerinin değişikliklerle ilgili hatalara neden olması problemini tanımlamak için kullanılan regresyon teriminden farklı olmasıdır.

3.1.2 Denetimsiz Öğrenme

Bu tür öğrenmede, algoritma eğitim aşamasında etiketlenmemiş verilerden MÖ modelini oluşturur. Etiketlenmemiş veriler, eğitim sırasında giriş verilerindeki kalıpları çıkarmak için algoritma tarafından kullanılır ve ortak özelliklerine göre girdileri farklı sınıflara atar. Test aşamasında ise, eğitilmiş model yeni ve görülmemiş bir veri setine uygulanır ve giriş verilerinin hangi sınıflara atanması gerektiğini tahmin eder. Çıktı doğruluk seviyesi tatmin edici bulunduğu model kullanıma alınır.

Denetimsiz öğrenme ile çözülen problemler iki kategoriye ayrılır:

- **Kümeleme:** Bu, problemde giriş veri noktalarındaki benzerliklerin tanımlanmasını gerektiren ve ortak özelliklere dayalı olarak gruplandırılmasına izin veren durumdur. Örneğin; kümeleme, pazarlama amacıyla farklı müşteri türlerini kategorize etmek için kullanılır.
- **İlişkilendirme:** Bu, problemin veri özellikleri arasında ilginç ilişkilerin veya bağımlılıkların belirlenmesini gerektirdiği durumdur. Örneğin; bir ürün öneri sistemi, müşterilerin alışveriş davranışlarına dayalı ilişkileri belirleyebilir.

3.1.3 Takviyeli Öğrenme

Takviyeli öğrenme, sistemin (bir "zeki ajan" olarak adlandırılan) çevre ile etkileşime girerek deneyimlerden öğrenme yaklaşımıdır. Takviyeli öğrenme, eğitim verisi kullanmaz. Ajan, doğru bir karar verdiğiğinde ödüllendirilir ve yanlış bir karar verdiğiğinde cezalandırılır.

Çevrenin kurulması, ajanın istenilen hedefe ulaşması için doğru stratejinin seçilmesi ve bir ödül fonksiyonunun tasarlanması, takviyeli öğrenmeyi uygularken karşılaşılan temel zorluklardır. Robotik, otonom araçlar ve sohbet botları, takviyeli öğrenmeyi kullanan uygulama örnekleridir.

3.2 MÖ İş Akışı

Makine Öğrenimi iş akışındaki faaliyetler şunlardır:

Hedefleri Anlamak

Dağıtılacak MÖ modelinin amacı, iş öncelikleri ile uyum sağlamak için paydaşlarla anlaşılmalı ve kabul edilmelidir. Geliştirilen model için kabul kriterleri (MÖ işlevsel performans ölçütleri dahil- bkz. Bölüm 5) belirlenmelidir.

Bir Çerçeve Seçmek

Uygun bir Yapay Zekâ geliştirme çerçevesi, hedefler, kabul kriterleri ve iş önceliklerine dayalı olarak seçilmelidir (bkz. Bölüm 1.5).

Algoritmayı Seçme ve Oluşturma

Bir MÖ algoritması, hedefler, kabul kriterleri ve mevcut veriler de dahil olmak üzere çeşitli faktörlere dayalı olarak seçilir (bkz. Bölüm 3.4). Algoritma genellikle önceden yazılmış kodların bulunduğu bir kütüphaneden alınır, ancak gerekiyorsa manuel olarak da kodlanabilir. Algoritma daha sonra modelin eğitimine hazırlanmak üzere derlenir.

Veri Hazırlığı ve Testi

Veri hazırlığı (bkz. Bölüm 4.1), veri edinimi, veri ön işleme ve özellik mühendisliğini içerir. Bu faaliyetlerle birlikte Keşifsel Veri Analizi (EDA) de yapılabilir.

Algoritma ve model tarafından kullanılan veriler, hedeflere dayanır ve Şekil 1'de gösterilen "model oluşturma ve test" faaliyetlerindeki tüm faaliyetlerde kullanılır. Örneğin, sistem bir gerçek zamanlı işlem sistemiyse, veriler işlem piyasasından gelecektir. Modeli eğitmek, ayarlamak ve test etmek için kullanılan veriler, model tarafından kullanılacak işletme verilerini temsil etmelidir.

Bazı durumlarda, modelin ilk eğitimi için önceden toplanmış veri setleri kullanmak mümkündür (örneğin, Kaggle veri setleri [R16]'ne bakınız). Aksi takdirde, ham veriler genellikle bazı ön işleme ve özellik mühendisliği gerektirir.

Verinin ve herhangi bir otomatik veri hazırlığı adımlarının test edilmesi gerekmektedir. Giriş verisi testi hakkında daha fazla bilgi için Bölüm 7.2.1'e bakınız.

Modeli Eğitmek

Seçilen MÖ algoritması, modeli eğitmek için eğitim verilerini kullanır.

Bazı algoritmalar, örneğin sinir ağı üretenler, eğitim veri setini birkaç kez okur. Eğitim veri setindeki her bir eğitim tekrarı bir dönem olarak adlandırılır.

Model yapısını tanımlayan parametreler (örneğin, bir sinir ağının katman sayısı veya bir karar ağacının derinliği) algoritmaya iletilir. Bu parametreler, model hiperparametreleri olarak bilinir.

Eğitimi kontrol eden parametreler (örneğin, bir sinir ağı eğitilirken kaç dönem kullanılacağı) de algoritmaya iletilir. Bu parametreler, algoritma hiperparametreleri olarak bilinir.

Modeli Değerlendirme

Model, onaylanmış MÖ işlevsel performans ölçütlerine karşı değerlendirilir, doğrulama veri seti kullanılarak ve sonuçlar daha sonra modeli geliştirmek (ayarlamak) için kullanılır. Model değerlendirmesi ve ayarlama, dikkatlice kontrol edilen koşullar altında, net belgelenmiş bir bilimsel deneyi andırmalıdır. Uygulamada, genellikle farklı algoritmalar kullanılarak birkaç model oluşturulur ve eğitilir (örneğin, rastgele ormanlar, SVM ve sinir ağları) ve değerlendirme ve ayarlama sonuçlarına dayanarak en iyi model seçilir.

Modeli Ayarlamak

Model, onaylanmış MÖ işlevsel performans ölçütlerine karşı değerlendirme sonuçları kullanılarak verilere uyacak şekilde model ayarlarının düzeltilmesi için kullanılır ve böylelikle performansı iyileştirilir. Model, hiperparametre ayarı ile ayarlanabilir, burada eğitim etkinliği değiştirilir (örneğin, eğitim adımlarının sayısını değiştirerek veya eğitim için kullanılan veri miktarını değiştirerek) veya modelin öznelikleri güncellenir (örneğin, bir sinir ağındaki nöron sayısı veya bir karar ağacının derinliği).

Eğitim, değerlendirme ve ayarlama olmak üzere üç faaliyet, Şekil 1'de gösterildiği gibi model oluşturmayı oluşturur.

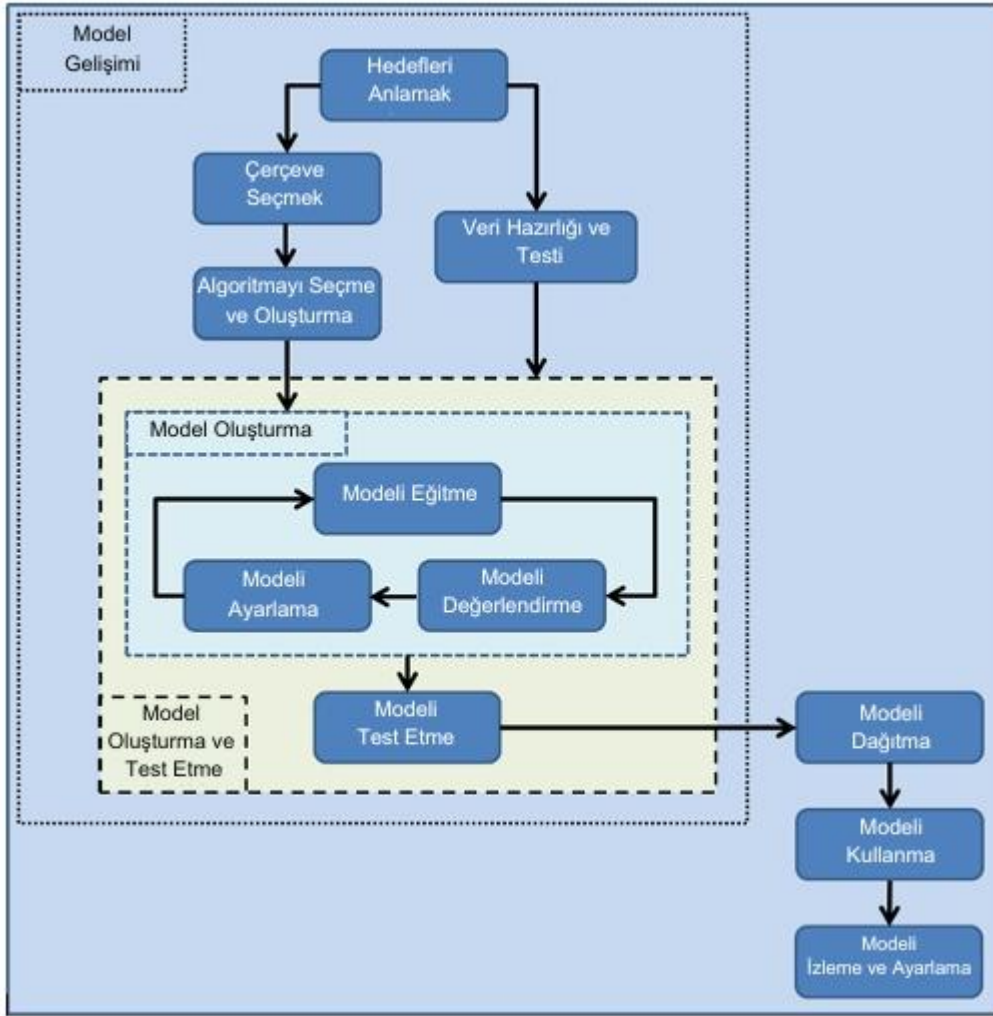
Modeli Test Etme

Bir model oluşturulduktan sonra (yani, eğitildi, değerlendirildi ve ayarlandı), kabul edilen MÖ işlevsel performans kriterlerinin karşılandığından emin olmak için bağımsız bir test veri setine karşı test edilmelidir (bkz. Bölüm 7.2.2). Testten elde edilen işlevsel performans ölçüleri, değerlendirmeden elde edilenlerle karşılaştırılır ve bağımsız veriyle modelin performansı değerlendirme sırasındaki performansından önemli ölçüde düşükse, farklı bir model seçmek gerekebilir.

İşlevsel performans testlerine ek olarak, modelin eğitim süresi, tahmin sunmak için gereken süre ve kaynak kullanımı gibi işlevsel olmayan testler de yapılmalıdır. Genellikle, bu testler veri mühendisi/bilim insanı tarafından gerçekleştirilir, ancak alanın yeterli bilgisine sahip ve ilgili kaynaklara erişimi olan testçiler de bu testleri gerçekleştirebilir.

Modeli Dağıtma

Model geliştirme tamamlandığında, Şekil 1'de gösterildiği gibi, genellikle ayarlanmış modelin ilgili veri akışıyla birlikte dağıtılması için yeniden mühendislik yapılması gerekmektedir. Bu genellikle çerçeve aracılığıyla gerçekleştirilir. Hedefler arasında gömülü sistemler ve bulut bulunabilir, burada model bir web API aracılığıyla erişilebilir.



Şekil 1: MÖ İş Akışı

Modeli Kullanma

Dağıtıldıktan sonra, model genellikle daha büyük bir Yapay Zekâ tabanlı sisteminin bir parçası olur ve operasyonel olarak kullanılabilir hale gelir. Modeller, belirlenen zaman aralıklarında zamanlanmış toplu tahminler yapabilir veya gerçek zamanlı olarak istek üzerine çalışabilir.

Modeli İzleme ve Ayarlama

Model kullanıldığı sürece, durumu gelişebilir ve model istenilen performansından uzaklaşabilir (bkz. Bölüm 2.3 ve 7.6). Herhangi bir sapmanın belirlenmesi ve yönetilmesi için, işletimsel modelin kabul kriterlerine göre düzenli olarak gözden geçirilmesi gerekmektedir.

Sapmanın giderilmesi için model ayarlarının güncellenmesi gerektiğinde ya da daha doğru veya daha sağlam bir model oluşturmak için yeni verilerle tekrar eğitimin gerektiği düşünülebilir. Bu durumda, güncellenmiş eğitim verileriyle yeni bir model oluşturulabilir ve eğitilebilir. Yeni model daha sonra var olan modelle A/B testi kullanılarak karşılaştırılabilir (bkz. Bölüm 9.4).

Şekil 1'de gösterilen MÖ iş akışı mantıksal bir sıradır. Uygulamada, iş akışı adımları tekrar tekrar döngüsel bir şekilde uygulanır (örneğin, model değerlendirildiğinde, genellikle eğitim adımına geri dönülmesi gerekebilir ve bazen de veri hazırlığına dönülmesi gerekebilir).

Şekil 1'de gösterilen adımlar, MÖ modelinin genel sistemdeki MÖ olmayan kısımlarla entegrasyonunu içermez. Tipik olarak, MÖ modelleri izole bir şekilde dağıtılamaz ve MÖ olmayan kısımlarla entegre edilmeleri gerekir. Örneğin; görüş uygulamalarında, verileri MÖ modeline göndermeden önce temizleyen ve değiştiren bir veri hattı bulunmaktadır. Modelin daha büyük bir Yapay Zekâ tabanlı sistemde bir parça olduğu durumlarda, bu sisteme dağıtılmadan önce bu modelin entegre edilmesi gerekecektir. Bu durumda, entegrasyon, sistem ve kabul testi seviyeleri, Bölüm 7.2'de açıklandığı gibi gerçekleştirilebilir.

3.3 MÖ Türünü Seçme

Uygun bir MÖ yaklaşımı seçerken, aşağıdaki yönergeler geçerlidir:

- Seçilen MÖ yaklaşımı için yeterli eğitim ve test verisi bulunmalıdır.
- Denetimli öğrenme için uygun şekilde etiketlenmiş verilere ihtiyaç vardır.
- Bir çıktı etiketi varsa, bu denetimli öğrenme olabilir.
- Çıktı kategorik ve kesikli ise, bu sınıflandırma olabilir.
- Çıktı sayısal ve sürekli ise, bu regresyon olabilir.
- Verilen veri setinde çıktı sağlanmıyorsa, bu denetimsiz öğrenme olabilir.
- Sorun benzer verilerin gruplandırılmasını içeriyorsa, bu kümeleme olabilir.
- Sorun eş zamanlı veri öğelerini bulmayı içeriyorsa, bu ilişkilendirme olabilir.
- Takviyeli öğrenme, çevre ile etkileşimin olduğu bağlamlara daha uygun olabilir.
- Sorun birden fazla durum kavramını içeriyorsa ve her durumda kararlar alınması gerekiyorsa, o zaman takviyeli öğrenme uygulanabilir olabilir.

3.4 MÖ Algoritması Seçiminde Yer Alan Faktörler

Optimal MÖ algoritması, MÖ modeli ayarları ve MÖ modeli hiperparametrelerinin seçimi için kesin bir yaklaşım bulunmamaktadır. Uygulamada, bu set aşağıdaki faktörlerin bir karışımına dayanarak seçilir:

- Gerekli işlevsellik (örneğin; işlevsellik sınıflandırma mı yoksa kesikli bir değer tahmini mi gerektiriyor)
- Gerekli kalite özellikleri; aşağıdakiler gibi
 - doğruluk (örneğin; bazı modeller daha doğru olabilir, ancak daha yavaş olabilir)
 - mevcut bellek üzerindeki kısıtlamalar (örneğin; gömülü bir sistem için)
 - modelin eğitilme (ve yeniden eğitilme) hızı
 - tahmin hızı (örneğin; gerçek zamanlı sistemler için)
 - şeffaflık, yorumlanabilirlik ve açıklanabilirlik gereksinimleri
- Modelin eğitimi için mevcut veri türü (örneğin; bazı modeller yalnızca görüntü verileri ile çalışabilir)
- Modelin eğitimi ve testi için mevcut veri miktarı (bazı modeller, örneğin, sınırlı miktarda veri ile diğer modellere göre daha fazla aşırı uyum eğiliminde olabilir)
- Model tarafından kullanılması beklenen girdi verilerindeki özellik sayısı (örneğin; hız ve doğruluk gibi diğer faktörler, özellik sayısından doğrudan etkilenebilir)
- Kümeleme için beklenen sınıf sayısı (örneğin; bazı modeller birden fazla sınıfı olan sorunlar için uygun olmayabilir)
- Önceki deneyim
- Deneme yanılma

3.5 Aşırı Uyum ve Yetersiz Uyum

3.5.1 Aşırı Uyum (Overfitting)

Aşırı uyum, modelin veri noktalarına çok sıkı bir şekilde uyması ve uygun şekilde genelleme yapmaması durumunda ortaya çıkar. Bu tür bir model, eğitimde kullanılan verilerle çok iyi çalışır ancak yeni veriler için doğru tahminler sağlamakta zorlanabilir. Aşırı uyum, modelin her veri noktasına, gürültü veya aykırı değerler olarak tanımlanabilecek veri noktalarını da dahil etmeye çalıştığında ortaya çıkabilir. Ayrıca, eğitim veri setinde yetersiz veri sağlandığında da görülebilir.

3.5.2 Yetersiz Uyum (Underfitting)

Yetersiz uyum, modelin eğitim verisindeki desenlere doğru şekilde uymak için yeterince sofistike olmadığı durumlarda ortaya çıkar. Yetersiz uyumlu modeller genellikle çok basit olur ve hem yeni veriler hem de eğitim verilerine çok benzer veriler için doğru tahminler sağlamakta zorlanabilir. Yetersiz uyumun bir nedeni, girişler arasındaki önemli ilişkileri yansıtmayan özelliklere sahip bir eğitim veri setidir. Ayrıca, algoritmanın verilere doğru şekilde uymaması durumunda da olabilir (örneğin; doğrusal olmayan veriler için doğrusal bir model oluşturmak).

3.5.3 Uygulamalı Egzersiz: Aşırı Uyum ve Yetersiz Uyumun Gösterilmesi

Model üzerinde aşırı uyum ve yetersiz uyum kavramlarını gösterin. Bu, çok az veri içeren bir veri seti (aşırı uyum) ve zayıf özellik korelasyonlarına sahip bir veri seti (yetersiz uyum) kullanılarak gösterilebilir.

4. MÖ - Veri – 230 Dakika

Anahtar kelimeler

Yok

YZ-Spesifik Anahtar Kelimeleri

Etiketleme, Arttırma, Sınıflandırma Modeli, Veri Etiketleme, Veri Hazırlığı, Makine Öğrenimi Eğitim Verisi, Denetimli Öğrenme, Test Veri Seti, Doğrulama Veri Seti.

Bölüm 4 için Öğrenme Hedefleri:

4.1 MÖ İş Akışının bir parçası olarak Veri Hazırlama

YZ-4.1.1 K2 Veri hazırlama ile ilgili etkinlikleri ve zorlukları açıklayın.

HO-4.1.1 H2 MÖ modelinin oluşturulmasını desteklemek için veri hazırlığı gerçekleştirin.

4.2 MÖ iş akışında eğitim, doğrulama ve test veri setleri

YZ-4.2.1 K2 MÖ model geliştirme sürecinde eğitim, doğrulama ve test veri setlerinin kullanımını karşılaştırın.

HO-4.2.1 H2 Eğitim ve test veri setlerini belirleyin ve bir Makine Öğrenimi modeli oluşturun.

4.3 Veri Seti Kalitesi Sorunları

YZ-4.3.1 K2 Tipik veri seti kalite sorunlarını açıklayın.

4.4 Veri kalitesi ve bu kalitenin MÖ modeli üzerindeki etkisi

YZ-4.4.1 K2 Düşük veri kalitesinin, ortaya çıkan MÖ modelinde nasıl sorunlara yol açabileceğini anlayın.

4.5 Denetimli Öğrenme için Veri Etiketleme

YZ-4.5.1 K1 Denetimli öğrenme için veri setlerindeki verilerin etiketlenmesi için farklı yaklaşımları hatırlayın.

YZ-4.5.2 K1 Veri setlerindeki verilerin yanlış etiketlenme nedenlerini hatırlayın.

4.1 MÖ İş Akışının Bir Parçası Olarak Veri Hazırlama

Veri hazırlığı, MÖ iş akışındaki eforun ortalama %43'ünü oluşturur ve muhtemelen MÖ iş akışındaki en çok kaynak tüketen faaliyettir. Buna karşılık, model seçimi ve oluşturma sadece %17'sini kullanır. Veri hazırlığı, ham veriyi alan ve hem bir MÖ modelini eğitmek için hem de eğitilmiş bir MÖ modeli tarafından tahmin için kullanılacak bir biçimde veri çıktısı üreten veri hattının bir parçasını oluşturur.

Veri hazırlamanın aşağıdaki faaliyetlerden oluştuğu düşünülebilir:

Veri edinimi

- **Tanımlama:** Eğitim ve tahminler için kullanılacak veri türleri belirlenir. Örneğin; bir otonom araç için, bu, radar, video ve lazer görüntüleme, tespit ve uzaklık ölçümü (LiDAR) verilerinin gerekliliğinin belirlenmesini içerebilir.
- **Toplama:** Verinin kaynağı belirlenir ve verinin toplanması için yöntemler belirlenir. Örneğini; bu, Uluslararası Para Fonu (IMF) gibi bir finansal veri kaynağının belirlenmesini ve verinin Yapay Zekâ tabanlı sisteme aktarılması için kullanılacak kanalların belirlenmesini içerebilir.
- **Etiketleme:** Bkz. Bölüm 4.5.

Edinilen veri çeşitli biçimlerde olabilir (örneğin; sayısal, kategorik, görüntü, tablo, metin, zaman serisi, sensör, coğrafi, video ve ses).

Veri ön işleme

- **Temizleme:** Yanlış veri, yinelenen veri veya aykırı değerler tespit edildiğinde, bunlar ya kaldırılır ya da düzeltilir. Ayrıca, eksik veri değerlerini tahmini veya öngörülen değerlerle (örneğin, ortalama, medyan ve mod değerlerini kullanarak) değiştirmek amacıyla veri yükleme kullanılabilir. Kişisel bilgilerin kaldırılması veya anonim hale getirilmesi de gerçekleştirilebilir.
- **Dönüşüm:** Verilen verinin biçimi değiştirilir (örneğin, bir dize olarak tutulan bir adresin bileşen parçalarına ayrılması, rastgele bir tanımlayıcı tutan bir alanın atılması, kategorik verilerin sayısal verilere dönüştürülmesi, görüntü biçimlerinin değiştirilmesi). Sayısal veriler üzerinde uygulanan dönüşümlerden bazıları, aynı aralığın kullanılmasını sağlamak için ölçeklendirme içerir. Standartlaştırma örneğin, veriyi sıfır ortalamalı ve bir standart sapmaya sahip olacak şekilde yeniden ölçekler. Bu normalleştirme, verinin sıfır ile bir arasında bir aralığa sahip olduğunu sağlar.
- **Artırma:** Bu, bir veri setindeki örnek sayısını artırmak için kullanılır. Veri artırma, aynı zamanda eğitim verilerine karşı örnekler dahil etmek için de kullanılabilir ve bu da karşı saldırılara karşı dayanıklılık sağlar (bkz. 9.1).
- **Örnekleme:** Bu, genel olarak mevcut toplam veri setinin bir kısmının seçilmesini içerir, böylece daha büyük veri setindeki desenler gözlemlenebilir. Bu, genellikle maliyetleri ve Makine Öğrenimi modelini oluşturmak için gereken zamanı azaltmak için yapılır.

Tüm ön işlemlerin, yararlı geçerli verileri değiştirme veya geçersiz veriler ekleme riski taşıdığını unutmayın.

Özellik mühendisliği

- Özellik seçimi: Bir özellik, veride yansıtılan bir özellik / niteliktir. Özellik seçimi, model eğitime ve tahminine en fazla katkıda bulunması muhtemel olan özelliklerin seçimini içerir. Uygulamada, genellikle sonuçta model üzerinde herhangi bir etkisi olması beklenmeyen (veya istenmeyen) özelliklerin çıkarılmasını içerir. Gereksiz bilgileri (gürültüyü) çıkararak, özellik seçimi genel eğitim sürelerini azaltabilir, aşırı uyumunu önleyebilir (bkz. Bölüm 3.5.1), doğruluğu artırabilir ve modelleri daha genelleştirilebilir hale getirebilir.
- Özellik Çıkarımı: Bu, mevcut özelliklerden bilgilendirici ve gereksiz olmayan özelliklerin türetilmesini içerir. Elde edilen veri seti tipik olarak daha küçüktür ve daha ucuz ve daha hızlı bir şekilde eşdeğer doğrulukta bir Makine Öğrenimi (MÖ) modeli oluşturmak için kullanılabilir.

Bu veri hazırlığı faaliyetlerine paralel olarak, genellikle genel veri hazırlama görevini desteklemek için keşfedici veri analizi (EDA) de gerçekleştirilir. Bu, veride doğal olarak bulunan trendleri keşfetmek için veri analizi yapmayı ve veriyi görsel bir formatta temsil etmek için veri görselleştirmesini içerir, bu da verideki trendleri çizerek veriyi görsel olarak temsil eder.

Yukarıdaki veri hazırlığı faaliyetleri ve alt faaliyetleri mantıklı bir sıra içinde gösterilmiş olsa da farklı projeler bunları yeniden düzenleyebilir veya yalnızca bir alt kümesini kullanabilir. Veri kaynağının belirlenmesi gibi bazı veri hazırlığı adımları sadece bir kez gerçekleştirilir ve manuel olarak gerçekleştirilebilir. Diğer adımlar ise operasyonel veri boru hattının bir parçası olabilir ve genellikle canlı veriler üzerinde çalışır. Bu görevler otomatikleştirilmelidir.

4.1.1 Veri Hazırlığında Karşılaşılan Zorluklar

Veri hazırlığıyla ilgili bazı zorluklar şunlardır:

- Gereksinimler:
 - Uygulama alanının bilgisine ihtiyaç duyma.
 - Veri ve özellikleri hakkında bilgi sahibi olma.
 - Veri hazırlığı ile ilişkili çeşitli tekniklerin bilinmesi.
- Birden çok kaynaktan yüksek kaliteli veri elde etme zorluğu.
- Veri hattını otomatikleştirmenin zorluğu ve üretim veri hattının hem ölçeklenebilir hem de makul performans verimliliğine sahip olduğundan emin olma zorluğu (örneğin; bir veri ögesinin işlenmesini tamamlamak için gereken zaman).
- Veri hazırlığıyla ilişkilendirilen maliyetler.
- Veri hazırlığı sırasında veri hattına eklenen hataların kontrol edilmesine yeterli öncelik verilmemesi.
- Örneklem yanlılığının tanıtılması (bkz. Bölüm 2.4).

4.1.2 Uygulamalı Egzersiz: Makine Öğrenimi için Veri Hazırlığı

Verilen ham veri seti için, Denetimli Öğrenme kullanarak sınıflandırma modeli oluşturmak için Bölüm 4.1'de açıklanan ilgili veri hazırlığı adımlarını gerçekleştirin.

Bu faaliyet, gelecekteki egzersizlerde kullanılacak bir MÖ modeli oluşturmanın ilk adımını oluşturur.

Bu faaliyeti gerçekleştirmek için, öğrencilere uygun (ve dile özel) materyaller sağlanacaktır, bunlar:

- Kütüphaneler
- MÖ çerçeveleri
- Araçlar
- Bir geliştirme ortamı

4.2 MÖ İş Akışında Eğitim, Doğrulama ve Test Veri Setleri

Mantıksal olarak, bir MÖ modeli geliştirmek için eşdeğer üç veri seti (yani tek bir başlangıç veri setinden rastgele seçilmiş) gereklidir:

- Modeli eğitmek için kullanılan eğitim veri seti.
- Modelin değerlendirilmesi ve ardından ayarlanması için kullanılan doğrulama veri seti.
- Ayarlanmış modelin test edilmesi için kullanılan test veri seti (aynı zamanda tutulan veri seti olarak da bilinir).

Sınırsız uygun veri mevcut ise, eğitim, değerlendirme ve test için kullanılan veri miktarı genellikle aşağıdaki faktörlere bağlıdır:

- Modeli eğitmek için kullanılan algoritma.
- RAM, disk alanı, hesaplama gücü, ağ bant genişliği ve mevcut süre gibi kaynakların bulunabilirliği.

Uygulamada, yeterli uygun veri elde etme zorluğu nedeniyle, eğitim ve doğrulama veri setleri genellikle tek bir birleştirilmiş veri setinden türetilir. Test veri seti ayrı tutulur ve eğitim sırasında kullanılmaz. Bu, geliştirilen modelin test verisi tarafından etkilenmemesini sağlamak ve test sonuçlarının modelin kalitesinin gerçek bir yansımaları vermesini sağlamak içindir.

Kombine veri setinin üç ayrı veri setine bölünmesi için optimal bir oran yoktur, ancak genellikle bir kılavuz olarak kullanılacak tipik oranlar, eğitim: doğrulama: test için 60:20:20 ile 80:10:10 arasında değişmektedir. Verinin bu veri setlerine bölünmesi genellikle rastgele yapılır, ancak veri küçükse veya elde edilen veri setlerinin beklenen işletim verilerini temsil etmemesi riski varsa, bu durumda rastgele yapılmaz.

Sınırlı veri mevcut ise, mevcut veriyi üç veri setine bölmek etkili bir eğitim için yeterli verinin olmamasına neden olabilir. Bu sorunu aşmak için, eğitim ve doğrulama veri setleri birleştirilebilir (test veri seti ayrı tutularak), ardından bu veri setinin birden fazla bölme kombinasyonu oluşturulabilir (örneğin, %80 eğitim / %20 doğrulama). Veri daha sonra eğitim ve doğrulama veri setlerine rastgele atanır. Bu çoklu bölme kombinasyonları kullanılarak eğitim, doğrulama ve ayarlama gerçekleştirilir ve birden çok ayarlanmış model oluşturulur, genel model performansı tüm çalışmaların ortalaması olarak hesaplanabilir. Birden fazla bölme kombinasyonu oluşturmak için kullanılan çeşitli yöntemler vardır. Bu yöntemler arasında bölme testi (split-test), ön yükleme (bootstrap), K-katlamalı çapraz doğrulama ve birini bırak çapraz doğrulama (leave-one-out cross validation) bulunmaktadır (daha fazla bilgi için bkz. [B02]).

4.2.1 Uygulamalı Egzersiz: Eğitim ve Test Verilerini Tanımlama ve Bir MÖ Modeli Oluşturma

Önceden hazırlanan veriyi (bkz. Bölüm 4.1.2) eğitim, doğrulama ve test veri setlerine ayırın.

Bu veri setleriyle denetimli öğrenme kullanarak bir sınıflandırma modeli eğitin ve test edin.

Doğrulama ve test veri setleri ile elde edilen doğruluk değerlerini karşılaştırarak, değerlendirme/ayarlama ile test etme arasındaki farkı açıklayın.

4.3 Veri Seti Kalite Sorunları

Bir veri setindeki verilerle ilgili tipik kalite sorunları şunları içerebilir, ancak bunlarla sınırlı değildir:

Kalite Özelliği	Açıklama
Yanlış veri	Yakalanan veri yanlış olabilir (örneğin, hatalı bir sensör aracılığıyla) veya yanlış girilmiş olabilir (örneğin, kopyala-yapıştır hataları).
Eksik veri	Veri değerleri eksik olabilir (örneğin, bir kayıta bir alan boş olabilir veya belirli bir zaman aralığı için veriler atlanmış olabilir). Eksik verilerin çeşitli nedenleri olabilir, bunlar arasında güvenlik sorunları, donanım sorunları ve insan hataları bulunabilir.
Hatalı etiketlenmiş veri	Veriler çeşitli nedenlerle yanlış etiketlenebilir. (bkz. Bölüm 4.5.2).
Yetersiz veri	Kullanılan öğrenme algoritmaları tarafından desenlerin tanınması için yeterli veri mevcut değildir (farklı algoritmalar için minimum gereken veri miktarı değişebilir).
Önceden işlenmemiş veriler	Verinin temiz olduğundan, tutarlı bir formatta olduğundan ve istenmeyen aykırı değerler içermediğinden emin olmak için verinin ön işlenmesi gerekmektedir (bkz. Bölüm 4.1).
Eskimiş veri	Öğrenme ve tahmin için kullanılan veriler mümkün olduğunca güncel olmalıdır (örneğin, birkaç yıl öncesine ait finansal verilerin kullanılması muhtemelen yanlış sonuçlara yol açabilir).
Dengesiz veri	Dengesiz veri, uygunsuz yanlılıktan (örneğin, ırk, cinsiyet veya etnik kökene dayalı), sensörlerin yanlış yerleştirilmesinden (örneğin, tavan yüksekliğine yerleştirilen yüz tanıma kameraları), veri setlerinin mevcudiyetindeki değişkenlikten ve veri tedarikçilerinin farklı motivasyonlarından kaynaklanabilir.
Adil Olmayan Veri	Adillik, öznel bir kalite özelliğidir ancak genellikle saptanabilir. Örneğin, çeşitliliği veya cinsiyet dengesini desteklemek için seçilen veriler, azınlıklara veya dezavantajlı gruplara karşı olumlu bir eğilime sahip olabilir (bu tür veriler adil kabul edilebilir ancak dengeli olmayabilir).
Yinelenen veriler	Tekrarlanan veri kayıtları, ortaya çıkan MÖ modelini gereğinden fazla etkileyebilir.
Alakasız veriler	Ele alınan sorunla ilgili olmayan veri, sonuçları olumsuz etkileyebilir ve kaynakların boşa harcanmasına neden olabilir.
Gizlilik sorunları	Herhangi bir veri kullanımı, ilgili veri gizliliği yasalarına (örneğin, Avrupa Birliği'ndeki bireylerin kişisel bilgileriyle ilgili olarak GDPR) uygun olmalıdır.
Güvenlik sorunları	Eğitim verilerine kasti olarak yerleştirilen sahte veya yanıltıcı veriler, eğitilmiş modelde doğruluk eksikliğine neden olabilir.

4.4 Veri Kalitesi ve MÖ Modeli Üzerindeki Etkisi

Makine Öğrenimi modelinin kalitesi, ondan oluşturulduğu veri setinin kalitesine son derece bağlıdır. Düşük kaliteli veri hem kusurlu modellere hem de kusurlu tahminlere yol açabilir.

Aşağıdaki hata kategorileri, veri kalitesi sorunlarından kaynaklanır:

- Azalmış doğruluk: Bu hatalar; yanlış, eksik, yanlış etiketlenmiş, yetersiz, eskimiş, ilgisiz ve ön işlenmemiş verilerden kaynaklanır. Örneğin; veri, beklenen ev fiyatları modelini oluşturmak için kullanıldığında, ancak eğitim verisi, kış bahçeli müstakil evler hakkında az veya hiç veri içermiyorsa, bu özel ev tipi için tahmin edilen fiyatların muhtemelen yanlış olacağı düşünülebilir.
- Yanlı model: Bu tür hatalar, verilerin eksik, dengesiz, adil olmayan, çeşitlilikten yoksun veya tekrarlanmış olmasından kaynaklanır. Örneğin, belirli bir özelliğe ait veriler eksikse (örneğin, hastalık tahmini için toplanan tüm tıbbi veriler yalnızca belirli bir cinsiyetten bireylerden toplanmışsa), bu durum ortaya çıkan model üzerinde olumsuz bir etkiye sahip olacaktır (model yalnızca o cinsiyet için tahmin yapmak amacıyla kullanılmayacaksa).
- Güvenliği ihlal edilmiş model: Bu hatalar, veri gizliliği ve güvenlik kısıtlamalarından kaynaklanmaktadır. Örneğin, verilerdeki gizlilik sorunları, saldırganların modellerden bilgileri tersine çevirmesine olanak tanır ve daha sonra kişisel bilgilerin sızmasına neden olabilecek güvenlik açıklarına yol açabilir.

4.5 Denetimli Öğrenme için Veri Etiketleme

Veri etiketleme, etiket eklenerek etiketlenmemiş (veya kötü etiketlenmiş) verinin zenginleştirilmesidir, böylece denetimli öğrenmede kullanılmaya uygun hale gelir. Veri etiketleme, ortalama olarak, MÖ projelerinde zamanın %25'ini kullandığı bildirilen yoğun kaynaklı bir faaliyettir [B11].

En basit haliyle, veri etiketleme, sınıflarına göre görüntüleri veya metin dosyalarını çeşitli klasörlere koymayı içerebilir. Örneğin; pozitif ürün incelemelerinin tüm metin dosyalarını bir klasöre ve negatif incelemeleri başka bir klasöre koymak gibi. Görüntülerdeki nesnelere çevreleyerek dikdörtgenler çizmek, genellikle bilinen bir diğer yaygın etiketleme tekniğidir ve sıkça "anotasyon" olarak adlandırılır. Daha karmaşık anotasyonlar, 3D nesnelere etiketlenmesi veya düzensiz nesnelere etrafına sınırlayıcı kutular çizmek için gereklidir. Veri etiketleme ve anotasyon genellikle araçlar tarafından desteklenir.

4.5.1 Veri Etiketleme Yaklaşımları

Etiketleme birkaç farklı şekilde gerçekleştirilebilir:

- Dahili: Etiketleme, geliştiriciler, test uzmanları veya etiketleme için kurulmuş bir organizasyon içindeki bir ekip tarafından gerçekleştirilir.
- Dış Kaynak Kullanımı: Etiketleme, harici uzman bir organizasyon tarafından yapılır.

- Kitle Kaynaklı: Etiketleme, geniş bir grup birey tarafından gerçekleştirilir. Etiketlemenin kalitesini yönetmenin zorluğu nedeniyle, birkaç etiketleyiciden aynı verileri etiketlemeleri istenebilir ve ardından kullanılacak etiket üzerine bir karar verilir.
- Yapay Zekâ Destekli: Yapay Zekâ tabanlı araçlar, verileri tanımlamak ve etiketlemek veya benzer verileri kümelemek için kullanılır. Sonuçlar daha sonra iki aşamalı bir sürecin parçası olarak bir insan tarafından onaylanır veya belki de tamamlanır (örneğin; sınırlayıcı kutuyu değiştirerek).
- Karma: Yukarıdaki etiketleme yaklaşımlarının bir kombinasyonu kullanılabilir. Örneğin, kitle kaynaklı etiketleme genellikle uzmanlaşmış Yapay Zekâ tabanlı kitle yönetimi araçlarına erişimi olan harici bir organizasyon tarafından yönetilir.

Uygun olduğu durumlarda, önceden etiketlenmiş bir veri setini yeniden kullanmak mümkün olabilir, böylece veri etiketlemeye tamamen gerek kalmaz. Birçok böyle veri seti genellikle, örneğin, Kaggle gibi kamuya açık kaynaklardan temin edilebilir [R16].

4.5.2 Veri Setlerinde Yanlış Etiketlenmiş Veriler

Denetimli öğrenme, verinin veri etiketleyiciler tarafından doğru bir şekilde etiketlendiğini varsayar. Ancak, bir veri setindeki tüm öğelerin doğru bir şekilde etiketlendiği uygulamada nadirdir. Veri aşağıdaki nedenlerle yanlış etiketlenebilir:

- Etiketleyiciler tarafından rastgele hatalar yapılabilir (örneğin, yanlış düğmeye basma).
- Sistemik hatalar yapılabilir (örneğin, etiketleyicilere yanlış talimatlar verilir veya yetersiz eğitim verilir).
- Kötü niyetli veri etiketleyicileri tarafından kasıtlı hatalar yapılabilir.
- Çeviri hataları, doğru bir şekilde etiketlenmiş veriyi bir dilde alır ve başka bir dilde yanlış etiketler.
- Yorumlamaya açık olan durumlarda, veri etiketleyicileri tarafından yapılan öznel değerlendirmeler farklı etiketlerin ortaya çıkmasına neden olabilir.
- Gerekli alan bilgisinin eksikliği yanlış etiketlemeye yol açabilir.
- Karmaşık sınıflandırma görevleri daha fazla hata yapılmasına neden olabilir.
- Veri etiketlemeyi desteklemek için kullanılan araçlar yanlış etiketlere yol açan hatalara sahip olabilir.
- MÖ tabanlı etiketleme yaklaşımları olasılıksaldır ve bu bazı yanlış etiketlere yol açabilir.

5. MÖ Fonksiyonel Performans Metrikleri - 120 dakika

Anahtar Kelimeler

Yok

YZ-Spesifik Anahtar Kelimeleri

Doğruluk, Eğri Altında Alan (AUC), Karışıklık Matrisi, F1-Skoru, Kümeler Arası Metrikler, Küme İçi Metrikler, Ortalama Kare Hatası (MSE), MÖ Kıyaslama Paketleri, MÖ Fonksiyonel Performans Metrikleri, Hassasiyet, Hatırlama, Alıcı İşletim Karakteristiği (ROC) Eğrisi, Regresyon Modeli, R-Kare, Siluet Katsayısı

Bölüm 5 için Öğrenme Hedefleri:

5.1 Karışıklık Matrisi

YZ-5.1.1 K3 Verilen bir karışıklık matrisi setinden MÖ fonksiyonel performans metriklerini hesaplayın.

5.2 Sınıflandırma, Regresyon ve Kümeleme için Ek MÖ Fonksiyonel Performans Metrikleri

YZ-5.2.1 K2 Sınıflandırma, regresyon ve kümeleme yöntemleri için MÖ fonksiyonel performans metriklerinin kavramlarını karşılaştırın ve karşılaştırın.

5.3 MÖ Fonksiyonel Performans Metriklerinin Sınırlamaları

YZ-5.3.1 K2 MÖ sistemlerinin kalitesini belirlemek için MÖ fonksiyonel performans metriklerinin kullanımının sınırlamalarını özetleyin.

5.4 MÖ Fonksiyonel Performans Metrikleri Seçimi

YZ-5.4.1 K4 Belirli bir MÖ modeli ve senaryo için uygun MÖ fonksiyonel performans metriklerini ve/veya değerlerini seçin.

HO-5.4.1 H2 Seçilen MÖ fonksiyonel performans metriklerini kullanarak oluşturulan MÖ modelini değerlendirin.

5.5 MÖ için Kıyaslama Paketleri

YZ-5.5.1 K2 MÖ bağlamında kıyaslama paketlerinin kullanımını açıklayın

5.1 Karışıklık Matrisi

Bir sınıflandırma probleminde, bir modelin sonuçları her zaman doğru tahmin etmesi nadirdir. Bu tür herhangi bir problem için, aşağıdaki olasılıklarla bir karışıklık matrisi oluşturulabilir:

		Güncel	
		Pozitif	Negatif
Tahmin	Pozitif	Gerçek Pozitif (TP)	Yanlış Pozitif (FP)
	Negatif	Yanlış Negatif (FN)	Doğru Negatif (TN)

Şekil 2: Karışıklık Matrisi

Şekil 2'de gösterilen karışıklık matrisinin farklı sunum şekilleri olabilir ancak her zaman dört olası durum olan gerçek pozitif (TP), gerçek negatif (TN), yanlış pozitif (FP) ve yanlış negatif (FN) için değerler üretecektir.

Karışıklık matrisine dayanarak, aşağıdaki metrikler tanımlanmıştır:

- Doğruluk

$$\text{Doğruluk} = (TP + TN) / (TP + TN + FP + FN) * 100\%$$

Doğruluk, tüm doğru sınıflandırmaların yüzdesini ölçer.

- Hassasiyet

$$\text{Hassasiyet} = TP / (TP + FP) * 100\%$$

Hassasiyet, doğru tahmin edilen pozitiflerin oranını ölçer. Pozitif tahminlere ne kadar emin olunabileceğinin bir ölçüsüdür.

- Hatırlama

$$\text{Hatırlama} = TP / (TP + FN) * 100\%$$

Hatırlama (aynı zamanda duyarlılık olarak bilinir), doğru tahmin edilen gerçek pozitiflerin oranını ölçer. Pozitifleri kaçırmamak konusunda ne kadar emin olunabileceğinin bir ölçüsüdür.

- F1-skoru

$$\text{F1-skoru} = 2 * (\text{Hassasiyet} * \text{Hatırlama}) / (\text{Hassasiyet} + \text{Hatırlama})$$

F1-skoru, hassasiyet ve hatırlamanın harmonik ortalaması olarak hesaplanır. Değeri sıfır ile bir arasında olacaktır. Bir'e yakın bir skor, yanlış verilerin sonuç üzerinde az etkisi olduğunu gösterir. Düşük bir F1-skoru, modelin pozitifleri tespit etmede zayıf olduğunu gösterir.

5.2 Sınıflandırma, Regresyon ve Kümeleme için Ek MÖ Fonksiyonel Performans Metrikleri

Çeşitli MÖ problemleri için farklı metrikler bulunmaktadır (Bölüm 5.1'de açıklananlar dışında). En yaygın kullanılan metriklerden bazıları aşağıda açıklanmıştır.

Denetimli Sınıflandırma Metrikleri

- Alıcı işletim karakteristiği (ROC) eğrisi, ikili bir sınıflandırıcının ayırım eşliğinin değiştirilmesiyle ilgili olarak grafiğe dökülen bir grafik çizimidir. Yöntem aslen askeri radarlar için geliştirilmiştir, bu yüzden bu şekilde adlandırılmıştır. ROC eğrisi, gerçek pozitif oranı (TPR) (aynı zamanda hatırlama olarak da bilinir) ve yanlış pozitif oranı ($FPR = FP / (TN + FP)$) ile çizilir, TPR y eksenini ve FPR x eksenini üzerinde.
- Eğri altında alan (AUC), ROC eğrisinin altındaki alanı temsil eder. Bir sınıflandırıcının ayırt ediciliğini gösterir, modelin sınıfları ne kadar iyi ayırt ettiğini gösterir. Daha yüksek bir AUC ile, modelin tahminleri daha iyidir.

Denetimli Regresyon Metrikleri

Denetimli regresyon modelleri için, metrikler regresyon çizgisinin gerçek veri noktalarına ne kadar iyi uydurduğunu temsil eder.

- Ortalama Kare Hatası (MSE), gerçek değer ve tahmin edilen değer arasındaki karelerin ortalamasıdır. MSE değeri her zaman pozitifdir ve sifıra daha yakın bir değer, daha iyi bir regresyon modelini önerir. Farkı karesini alarak, pozitif ve negatif hataların birbirini iptal etmesini sağlar.
- R-kare (aynı zamanda belirlilik katsayısı olarak da bilinir), regresyon modelinin bağımlı değişkenlere ne kadar iyi uyduğunu ölçer.

Denetimsiz Kümeleme Metrikleri

Denetimsiz kümeleme için, farklı kümeler arasındaki mesafeleri ve bir verilen kümedeki veri noktalarının yakınlığını temsil eden birkaç metrik vardır.

- Küme içi metrikler, bir küme içindeki veri noktalarının benzerliğini ölçer.
- Kümeler arası metrikler, farklı kümelerdeki veri noktalarının benzerliğini ölçer.
- Siluet katsayısı (aynı zamanda siluet skoru olarak da bilinir), ortalama kümeler arası ve küme içi mesafelere dayalı bir ölçüdür (-1 ile +1 arasında). +1 skoru, kümelerin iyi ayrıldığını, sıfır skoru rastgele kümeleme olduğunu ve -1 skoru kümelerin yanlış atandığını gösterir.

5.3 MÖ Fonksiyonel Performans Metriklerinin Sınırlamaları

MÖ fonksiyonel performans metrikleri, modelin işlevselliğini (örneğin doğruluk, hassasiyet, hatırlama, MSE, AUC ve siluet katsayısı açısından) ölçmekle sınırlıdır. Bunlar, ISO 25010'da tanımlanan diğer işlevsel olmayan kalite özelliklerini (örneğin, performans verimliliği) ve 2. bölümde açıklananları ölçmezler (örneğin, açıklanabilirlik, esneklik ve otonomi). Bu müfredatta, terim "MÖ fonksiyonel performans metrikleri" kullanılmıştır çünkü "performans metrikleri" teriminin bu işlevsel metrikleri ifade etmek için yaygın olarak kullanılması. "MÖ fonksiyonel" ifadesi, bu metriklerin Makine Öğrenimiyle ilgili olduğunu ve performans verimliliği metrikleriyle ilişkisi olmadığını vurgular.

MÖ fonksiyonel performans metrikleri birkaç diğer faktör tarafından kısıtlanmıştır:

- Denetimli öğrenme için, MÖ fonksiyonel performans metrikleri etiketli verilere dayalı olarak hesaplanır ve elde edilen metriklerin doğruluğu doğru etiketlemeye bağlıdır (bkz. Bölüm 4.5).
- Ölçüm için kullanılan veriler temsilci olmayabilir (örneğin, yanlış olabilir) ve oluşturulan MÖ fonksiyonel performans metrikleri bu verilere bağlıdır (bkz. Bölüm 2.4).
- Sistem birkaç bileşenden oluşabilir, ancak MÖ fonksiyonel performans metrikleri yalnızca MÖ modeline uygulanır. Örneğin, veri boru hattı, modeli değerlendirmek için MÖ fonksiyonel performans metrikleri tarafından dikkate alınmaz.
- MÖ fonksiyonel performans metriklerinin çoğu, araçların desteği olmadan ölçülemez.

5.4 MÖ Fonksiyonel Performans Metriklerinin Seçimi

Karışıklık matrisinden üretilen MÖ fonksiyonel performans metriklerinin tümü için en yüksek puanı alan bir MÖ modeli oluşturmak genellikle mümkün değildir. Bunun yerine, modelin beklenen kullanımına dayanarak kabul kriterleri olarak en uygun MÖ fonksiyonel performans metrikleri seçilir (örneğin, yanlış pozitifleri en aza indirmek için yüksek bir hassasiyet değeri gereklidir, yanlış negatifleri en aza indirmek için hatırlama metriği yüksek olmalıdır). 5.1 ve 5.2 bölümlerinde açıklanan MÖ fonksiyonel performans metriklerini seçerken aşağıdaki kriterler kullanılabilir:

- Doğruluk: Bu metrik, veri setlerinin simetrik olduğu durumlarda uygulanabilir olabilir (örneğin, yanlış pozitif ve yanlış negatif sayıları ve maliyetleri benzerdir). Bu metrik, bir veri sınıfının diğerlerinden daha baskın olduğu durumlarda kötü bir seçim haline gelir, bu durumda F1-skoru dikkate alınmalıdır.
- Hassasiyet: Yanlış pozitiflerin maliyeti yüksekse ve pozitif sonuçlara olan güven yüksekse bu uygun bir metrik olabilir. Örneğin, (bir e-postayı spam olarak sınıflandırma pozitif olarak kabul edilirse), spam filtresi yüksek hassasiyet gerektiren bir örnektir, çünkü aslında spam olmayan çok fazla e-postayı spam klasörüne koymak çoğu kullanıcı için kabul edilemez olacaktır. Sınıflandırıcı, durumlarla başa çıktığında, çok büyük bir yüzde olarak vakaların pozitif olduğu durumlar, yalnızca hassasiyet kullanmak muhtemelen iyi bir seçim olmayacaktır.
- Hatırlama: Pozitiflerin kaçırılmaması kritikse, yüksek bir hatırlama skoru önemlidir. Örneğin, kanser tespitinde herhangi bir gerçek pozitifin kaçırılması ve bunların negatif olarak işaretlenmesi (yani, kanser tespit edilmedi) kabul edilemez olabilir.
- F1-skoru – F1-skoru, beklenen sınıflar arasında bir dengesizlik olduğunda ve hassasiyet ile hatırlamanın benzer öneme sahip olduğu durumlarda en kullanışlıdır. Yukarıdaki metriklerin yanı sıra, Bölüm 5.2'de açıklanan birkaç metrik belirli MÖ problemleri için uygun olabilir, örneğin:
 - ROC eğrisi için AUC, denetimli sınıflandırma problemleri için kullanılabilir.
 - MSE ve R-kare, denetimli regresyon problemleri için kullanılabilir.
 - Kümeler arası metrikler, küme içi metrikler ve siluet katsayısı, denetimsiz kümeleme problemleri için kullanılabilir.

5.4.1 Uygulamalı Egzersiz: Oluşturulan Makine Öğrenme (MÖ) Modelini Değerlendirme

Önceki egzersizde eğitilen sınıflandırma modelini kullanarak, doğruluk, hassasiyet, hatırlama ve F1-skoru değerlerini hesaplayın ve görüntüleyin. Uygulanabilirse, hesaplamaları yapmak için geliştirme çerçeveniz tarafından sağlanan kütüphane işlevlerini kullanın.

5.5 MÖ için Kıyaslama Paketleri

Yeni YZ teknolojileri, düzenli olarak yeni veri setleri, algoritmalar, modeller ve donanımlar yayınlanmaktadır ve her birinin göreceli etkililiğini belirlemek zor olabilir.

Bu farklı teknolojiler arasında objektif karşılaştırmalar sağlamak için endüstri standardı MÖ kıyaslama paketleri bulunmaktadır. Bunlar geniş bir uygulama alanını kapsar ve YZ ve MÖ performansı için donanım platformları, yazılım çerçeveleri ve bulut platformlarını değerlendirmek için araçlar sağlar.

MÖ kıyaslama paketleri, eğitim süreleri (örneğin, bir çerçevenin belirli bir eğitim veri setini kullanarak bir MÖ modelini belirli bir hedef kalite metriği, örneğin %75 doğruluk, ile ne kadar hızlı eğitebileceğini) ve çıkarma süreleri (örneğin, eğitilen bir MÖ modelinin çıkarım yapma hızı) gibi çeşitli ölçümler sağlayabilir.

MÖ karşılaştırma paketleri, aşağıdakiler gibi birkaç farklı kuruluş tarafından sağlanmaktadır:

- MLCommons [R18]: Bu, 2020'de kurulmuş ve daha önce adlandırılmış, kar amacı gütmeyen bir kuruluştur.
- ML Perf, yazılım çerçeveleri, yapay zekaya özgü işlemciler ve makine öğrenimi bulut platformları için performans karşılaştırma ölçütleri sağlayan bir araçtır.
- DAWNBench [R19]: Bu, Stanford Üniversitesi'nden bir MÖ kıyaslama paketidir.
- MLMark [R20]: Bu, Gömülü Mikroişlemci Karşılaştırma Konsorsiyumu tarafından tasarlanan ve gömülü çıkarımın performansını ve doğruluğunu ölçmeyi amaçlayan bir Makine Öğrenimi kıyaslama paketidir.

6. MÖ - Yapay Sinir Ağları ve Test Edilmesi - 65 dakika

Anahtar kelimeler

Yok

YZ-Spesifik Anahtar Kelimeleri

Aktivasyon Değeri, Derin Sinir Ağı (DNN), MÖ Eğitim Verisi, Çok Katmanlı Algılayıcı, Sinir Ağı, Nöron Kapsamı, Perseptron (Algılayıcı), İşaret Değişim Kapsamı, İşaret-İşaret Kapsamı, Denetimli Öğrenme, Eşik Kapsamı, Eğitim Verisi, Değer Değişim Kapsamı.

Bölüm 6 için öğrenme hedefleri:

6.1 Yapay Sinir Ağları

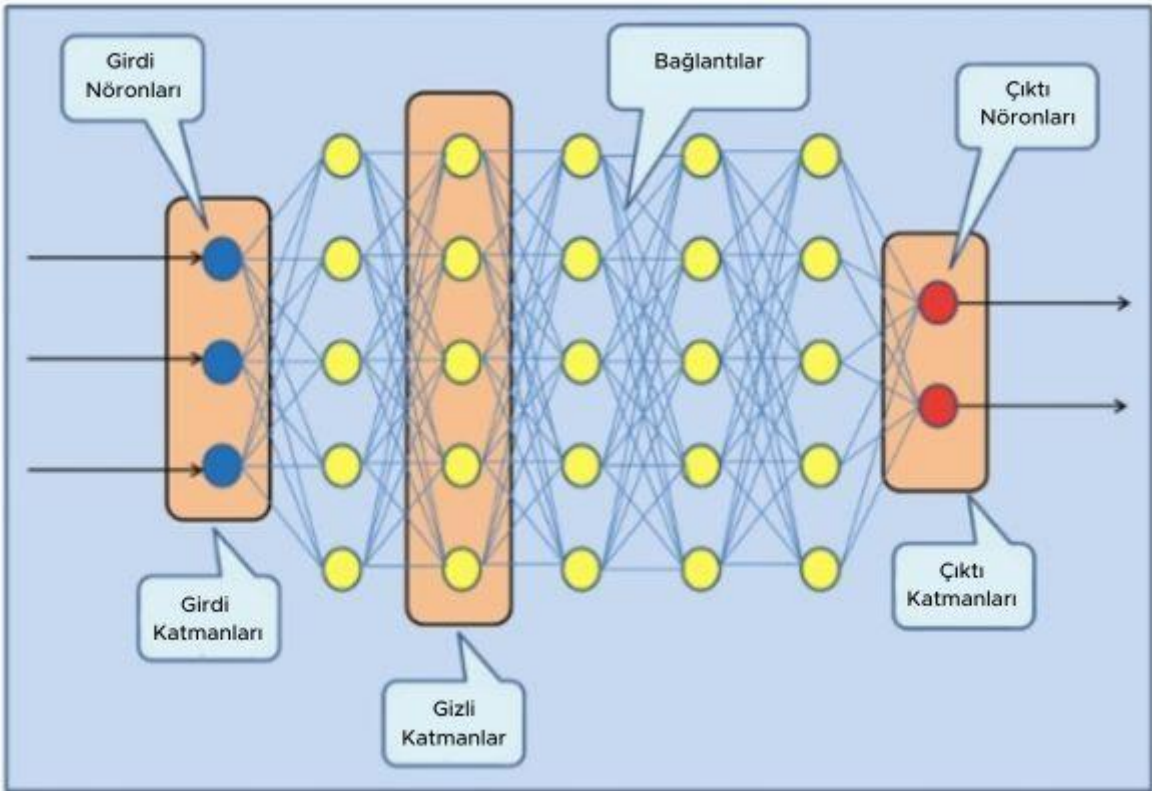
- YZ-6.1.1 K2 Bir DNN (Derin Sinir Ağı) içeren sinir ağının yapısını ve işlevini açıklayınız.
HO-6.1.1 H1 Bir perseptron (algılayıcı) uygulanmasını deneyimleyin.

6.2 Sinir Ağları için Kapsama Ölçüleri

- YZ-6.2.1 K2 Sinir ağları için farklı kapsama ölçümlerini açıklayınız

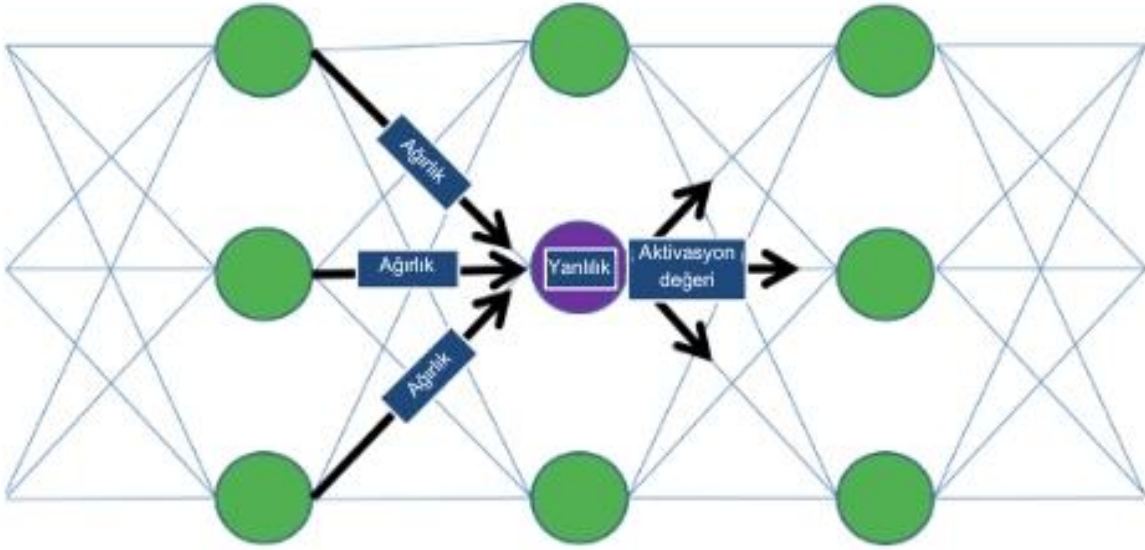
6.1 Yapay Sinir Ağları

Yapay sinir ağları başlangıçta insan beyninin işleyişini taklit etmek amacıyla tasarlanmıştır, ki bu, birbirine bağlı birçok biyolojik nöron olarak düşünülebilir. Tek katmanlı perseptron (algılayıcı), yapay sinir ağı uygulamalarının ilk örneklerden biridir ve sadece bir katmana (yani tek bir nörona) sahip bir sinir ağından oluşur. Bu, belirli bir sınıfa ait olup olmadığını belirleyen sınıflandırıcıların denetimli öğrenimi için kullanılabilir. Çoğu güncel sinir ağı, birden çok katmana sahip oldukları ve çok katmanlı algılayıcılar olarak düşünülebilecekleri için derin sinir ağları olarak kabul edilir (Bkz. Şekil 3).



Şekil 3: Derin sinir ağının yapısı

Derin sinir ağı, üç tür katmandan oluşur. Girdi katmanı, örneğin bir kameradan piksel değerleri gibi girdileri alır. Çıktı katmanı, sonuçları dış dünyaya iletir. Bu örneğin, girdi görüntüsünün bir kedi olma olasılığını belirten bir değer olabilir. Girdi ve çıktı katmanları arasında, yapay sinir hücrelerinden oluşan gizli katmanlar bulunur. Bu hücreler aynı zamanda düğüm olarak da bilinir. Bir katmandaki sinir hücreleri, bir sonraki katmandaki her bir sinir hücresine bağlanır ve ardışık katmanlarda farklı sayıda sinir hücresi olabilir. Sinir hücreleri hesaplamalar yapar ve bilgiyi giriş sinir hücrelerinden çıktı sinir hücrelerine ağı boyunca iletir.



Sekil 4: Her sinir hücresi tarafından gerçekleştirilen hesaplama

Şekil 4'te gösterildiği gibi, her bir nöronun (girdi katmanındaki nöronlar hariç) gerçekleştirdiği hesaplama, aktivasyon değeri olarak bilinen değeri üretir. Bu değer, önceki katmandaki tüm nöronların aktivasyon değerlerini, nöronlar arasındaki bağlantılara atanan ağırlıkları (bu ağırlıklar ağ öğrenirken değişir) ve her bir nöronun bireysel yanlılık değerini girdi olarak alan bir formülü (aktivasyon fonksiyonu) çalıştırarak hesaplanır. Bu yanlılık, önceden belirlenmiş sabit bir değerdir ve 2.4 Bölümünde ele alınan yanlılık ile ilgili değildir. Farklı aktivasyon fonksiyonlarını çalıştırmak, farklı aktivasyon değerlerinin hesaplanmasına neden olabilir. Bu değerler genellikle sıfır etrafında toplanır ve -1 ile +1 arasında bir aralığa sahiptir (-1 değeri nöronun 'ilgisiz' olduğunu, +1 değeri ise nöronun 'çok ilgili' olduğunu ifade eder).

Yapay sinir ağını eğitirken, her nörona önceden belirlenmiş bir yanlılık değeri atanır ve eğitim verisi ağdan geçirilir. Her nöron aktivasyon fonksiyonunu çalıştırarak nihayetinde bir çıktı üretir. Üretilen çıktı, bilinen doğru sonuçla (bu örnekte etiketli veri kullanılarak) karşılaştırılır. Gerçek çıktı ile bilinen doğru sonuç arasındaki fark, bu farkı minimize etmek amacıyla nöronlar arasındaki bağlantılardaki ağırlık değerlerini değiştirmek için ağa geri beslenir. Ağdan daha fazla eğitim verisi geçirildikçe, ağ öğrenirken ağırlıklar kademeli olarak ayarlanır. Sonuç olarak, üretilen çıktılar yeterince iyi kabul edildiğinde eğitim sona erer.

6.1.1 Uygulamalı Egzersiz: Basit Bir Perseptron'u (Algılayıcı) Uygulama

Öğrencilere, perseptron'un basit bir işlevi öğrenme sürecini gösteren, örneğin AND işlevi gibi, bir egzersiz yapılacaktır.

Egzersiz, perseptron'un ağırlıkları birkaç dönem boyunca hata sıfıra ulaşana kadar değiştirerek nasıl öğrendiğini kapsmalıdır. Bu etkinlik için çeşitli mekanizmalar kullanılabilir (örneğin, elektronik tablo, simülasyon).

6.2 Sinir Ağları için Kapsama Ölçüleri

Beyaz kutu test kapsamı kriterlerine (örneğin, deyim, dal, değiştirilmiş koşul/karar kapsamı (MC/DC) [I01]) ulaşmak, geleneksel emir kipi kaynak kodu kullanırken bazı güvenlikle ilgili standartlara uyum sağlamak için zorunludur [S07] ve diğer kritik uygulamalar için birçok test uzmanı tarafından tavsiye edilmektedir. Kapsamın izlenmesi ve iyileştirilmesi, yeni test senaryolarının tasarımını destekler ve test edilen nesneye olan güveni artırır.

Sinir ağlarının kapsamını ölçmek için bu tür önlemlerin kullanılması, genellikle her bir sinir ağı çalıştırıldığında aynı kodun çalıştırılması nedeniyle az bir değer sağlar. Bunun yerine, kapsama ölçüleri, sinir ağının yapısının ve daha spesifik olarak içindeki sinir hücrelerinin kapsamına dayalı olarak önerilmiştir. Bu ölçümlerin çoğu, sinir hücrelerinin aktivasyon değerlerine dayanmaktadır.

Sinir ağları için kapsama, yeni bir araştırma alanıdır. Akademik makaleler yalnızca 2017'den beri yayımlanmıştır ve bu nedenle, önerilen ölçümlerin etkili olduğunu gösteren nesnel kanıtlar (örneğin, tekrarlanan araştırma sonuçları) çok azdır. Ancak, ifade ve karar kapsamının 50 yılı aşkın bir süredir kullanılmasına rağmen, tıbbi cihazlar ve havacılık sistemleri gibi güvenlikle ilgili uygulamalardaki yazılım kapsamının ölçümü için zorunlu olmasına rağmen, göreceli etkinliklerine dair de çok az nesnel kanıt vardır.

Aşağıdaki kapsama kriterleri, araştırmacılar tarafından çeşitli uygulamalara önerilmiş ve uygulanmıştır:

- **Nöron kapsamı:** Tam nöron kapsamı, sinir ağındaki her bir nöronun bir aktivasyon değeri elde etmesini gerektirir ve bu değer sıfırdan büyük olması gerekir [B12]. Bu, pratikte çok kolay bir şekilde elde edilir ve araştırmalar, çeşitli derin sinir ağlarında çok az sayıda test senaryosu ile neredeyse %100 kapsama sağlandığını göstermiştir. Bu kapsama ölçüsü, başarısız olduğunda bir uyarı sinyali olarak en faydalı olabilir.
- **Eşik kapsamı:** Tam eşik kapsamı, sinir ağındaki her bir nöronun belirli bir eşik değerinden büyük bir aktivasyon değeri elde etmesini gerektirir. DeepXplore çerçevesini oluşturan araştırmacılar aslında nöron kapsamının, duruma bağlı olarak değişebilecek bir eşik değerini aşan aktivasyon değeri temel alınarak ölçülmesi gerektiğini önermişlerdir. Bu beyaz-kutu yaklaşımını kullanarak binlerce yanlış köşe durum davranışını etkin bir şekilde bulduklarını rapor ettiklerinde, araştırmayı 0.75 eşik değeri ile gerçekleştirmişlerdir. Bu kapsama türü, diğer araştırmacıların bazılarının "nöron kapsamı" terimini sıfır eşikli nöron kapsamı anlamında kullandığı için, onu daha kolay ayırt etmek için burada yeniden adlandırılmıştır.
- **İşaret Değişimi Kapsamı:** Tam işaret değişimi kapsamı elde etmek için, test senaryolarının her nöronun hem pozitif hem de negatif aktivasyon değerlerine ulaşmasını sağlaması gerekir [B13].
- **Değer Değişimi Kapsamı:** Tam değer değişimi kapsamı elde etmek için, test senaryolarının her nöronun iki aktivasyon değerine ulaşmasını sağlaması gerekir; bu iki değer arasındaki farkın belirli bir değeri aşması gerekmektedir [B13].
- **İşaret-İşaret kapsamı:** Bu kapsama, bitişik katmanlardaki nöron çiftlerini ve aktivasyon değerlerinin aldığı işareti dikkate alır. Bir nöron çiftinin kapsam içinde kabul edilmesi için, birinci katmandaki bir nöronun işaretini değiştirmenin ikinci katmandaki nöronun işaretini değiştirdiğini gösteren bir test senaryosu gereklidir, ancak ikinci katmandaki diğer tüm nöronların işaretleri değişmeden kalmalıdır [B13]. Bu, imperatif (yordamsal) kaynak kodlar için MC/DC kapsamına benzer bir kavramdır.

Araştırmacılar, katmanlara dayalı (işaret değişimi kapsamından daha basit olsa da) ek kapsam ölçütleri hakkında raporlar sunmuş ve TensorFuzz aracında uygulanmış olan, komşu nöron kümelerindeki anlamlı değişiklikleri belirlemek için en yakın komşu algoritmalarını kullanan başarılı bir yaklaşımı hayata geçirmiştir

[B14].

7.YZ Tabanlı Sistemlerin Test Edilmesi Genel Bakışı - 115 dakika

Anahtar kelimeler

Girdi verisi testi, Makine Öğrenimi modeli testi

YZ-Spesifik Anahtar Kelimeleri

Yapay Zekâ Bileşeni, Otomasyon Yanlılığı, Büyük Veri, Kavram Kayması, Veri Akışı, Makine Öğrenimi İşlevsel Performans Metrikleri, Eğitim Verisi

Bölüm 7 için Öğrenme Hedefleri:

7.1 Yapay Zekâ Tabanlı Sistemlerin Özellikleri:

YZ-7.1.1 K2 Yapay Zekâ tabanlı sistemler için sistem spesifikasyonlarının test etmede nasıl zorluklar yaratabileceğini açıklayın.

7.2 Yapay Zekâ Tabanlı Sistemler için Test Seviyeleri

YZ-7.2.1 K2 Her test seviyesi için Yapay Zekâ tabanlı sistemlerin nasıl test edildiğini açıklayın.

7.3 Yapay Zekâ Tabanlı Sistemlerin Test Edilmesi İçin Test Verileri

YZ-7.3.1 K1 Yapay Zekâ tabanlı sistemlerin test edilmesini zorlaştırabilecek test verileriyle ilişkili faktörleri hatırlayın.

7.4 Yapay Zekâ Tabanlı Sistemlerde Otomasyon Yanlılığının Test Edilmesi

YZ-7.4.1 K2 Otomasyon yanlılığını açıklayın ve bu durumun test etmeyi nasıl etkilediğini belirtin.

7.5 YZ Bileşenlerinin Belgelendirilmesi

YZ-7.5.1 K2 Bir Yapay Zekâ bileşeninin belgelenmesini açıklayın ve belgelerin Yapay Zekâ tabanlı sistemlerin test edilmesini nasıl desteklediğini anlayın.

7.6 Kavram Kayması İçin Test Etme

YZ-7.6.1 K2 Eğitilmiş modelin kavram kaymasını ele almak için sık sık test edilme ihtiyacını açıklayın.

7.7 Bir Makine Öğrenimi Sistemi İçin Bir Test Yaklaşımı Seçme

YZ-7.7.1 K4 Bir senaryo verildiğinde, bir Makine Öğrenimi sistemi geliştirilirken izlenecek bir test yaklaşımını belirleyin.

7.1 YZ Tabanlı Sistemlerin Özellikleri

Sistem gereksinimleri ve tasarım özellikleri hem Yapay Zekâ tabanlı sistemler hem de geleneksel sistemler için eşit derecede önemlidir. Bu özellikler, test uzmanlarına gerçek sistem davranışının belirtilen gereksinimlerle uyumlu olup olmadığını kontrol etme imkanı sağlar. Ancak, spesifikasyonlar eksikse ve test edilebilirlikten yoksunsa, bu bir test oracle (test doğrulayıcı) problemi ortaya çıkarır (bkz. Bölüm 8.7).

YZ tabanlı sistemlerin spesifikasyonunun özellikle zor olabileceği birkaç neden vardır:

- Birçok YZ tabanlı sistem projesinde, gereksinimler yalnızca üst düzey iş hedefleri ve gerekli tahminler açısından belirtilir. Bunun nedeni, YZ tabanlı sistem geliştirme sürecinin keşif niteliğinde olmasıdır. Çoğu zaman, YZ tabanlı sistem projeleri bir veri setiyle başlar ve bu verilerden hangi tahminlerin elde edilebileceğini belirlemek amaçlanır. Bu durum, geleneksel bir projede gerekli mantığın en baştan belirlenmesi ile tezat oluşturur.
- YZ tabanlı sistemlerin doğruluğu, bağımsız testlerden elde edilen sonuçlar elde edilene kadar genellikle bilinmez. Keşif niteliğindeki geliştirme yaklaşımıyla birlikte, istenen kabul kriterleri belirlendiğinde uygulama zaten devam ettiği için, bu genellikle yetersiz spesifikasyonlara yol açar.
- Birçok YZ tabanlı sistemin olasılıksal doğası, bazı beklenen kalite gereksinimleri için toleransların belirtilmesini gerektirebilir, örneğin tahminlerin doğruluğu gibi.
- Sistem hedefleri, belirli bir işlevsellik sağlamak yerine insan davranışını yeniden üretmeyi gerektirdiğinde, bu genellikle değiştirmeyi amaçladığı insan faaliyetlerinden daha iyi veya onlarla aynı düzeyde olma temeline dayanan zayıf tanımlanmış davranış gereksinimlerine yol açar. Bu durum, yerini aldığı insanların yetenekleri büyük ölçüde farklılık gösterdiğinde, bir test doğrulayıcı tanımlamayı zorlaştırabilir.
- YZ'nin doğal dil tanıma, bilgisayarlı görü ve insanlarla fiziksel etkileşim gibi kullanıcı arayüzlerini uygulamak için kullanıldığı durumlarda, sistemlerin artan esneklik göstermesi gerekir. Ancak, bu esneklik, bu tür etkileşimlerin gerçekleşebileceği tüm farklı yolların tanımlanması ve belgelenmesinde zorluklar yaratabilir.
- Uyarlanabilirlik, esneklik, evrim ve otonomi gibi YZ tabanlı sistemlere özgü kalite özellikleri, gereksinimlerin belirlenmesinin bir parçası olarak düşünülüp tanımlanmalıdır (bkz. Bölüm 2). Bu özelliklerin yeniliği, onları tanımlamayı ve test etmeyi zorlaştırabilir.

7.2 YZ Tabanlı Sistemler için Test Seviyeleri

YZ tabanlı sistemler genellikle hem Yapay Zekâ hem de Yapay Zekâ olmayan bileşenler içerir. Yapay Zekâ olmayan bileşenler, geleneksel yaklaşımlar kullanılarak test edilebilirken [101], Yapay Zekâ bileşenlerinin ve bu bileşenleri içeren sistemlerin aşağıda açıklandığı gibi bazı yönlerden farklı bir şekilde test edilmesi gerekebilir. Yapay Zekâ bileşenlerinin test edildiği tüm test seviyeleri için, testin veri mühendisleri/bilim adamları ve alan uzmanları tarafından yakından desteklenmesi önemlidir.

Geleneksel yazılımlar için kullanılan test seviyelerinden önemli bir fark, YZ tabanlı sistemlerde kullanılan girdi verisinin ve modellerin açıkça test edilmesini ele almak için iki yeni uzmanlaşmış test seviyesinin dahil edilmesidir [B15]. Bu bölümün çoğu, tüm YZ tabanlı sistemlere uygundur, ancak bazı bölümler özellikle Makine Öğrenimine (MÖ) odaklanmıştır.

7.2.1 Girdi Verisi Testi

Girdi verisi testinin amacı, sistemin eğitim ve tahmin için kullandığı verinin en yüksek kalitede olduğunu sağlamaktır (bkz. Bölüm 4.3). Bu şunları içerir:

- İncelemeler
- İstatistiksel teknikler (örneğin, veriyi yanlılık açısından test etme)
- Eğitim verisinin EDA'sı (Keşifsel Veri Analizi)
- Veri hattının statik ve dinamik testi

Veri hattı genellikle veri hazırlama işlemlerini gerçekleştiren birkaç bileşeni içerir (bkz. Bölüm 4.1), ve bu bileşenlerin testi hem bileşen testi hem de entegrasyon testini içerir. Eğitim için veri hattı, operasyonel tahminleri desteklemek için kullanılan tamamen mühendislikle otomatikleştirilmiş sürüme kıyasla bir prototip olarak kabul edilebilir. Bu nedenle, bu iki sürümün veri hattının testi oldukça farklı olabilir. Ancak, iki sürümün fonksiyonel eşdeğerliğinin test edilmesi de dikkate alınmalıdır.

7.2.2 MÖ Model Testi

Makine Öğrenimi modeli testinin amacı, belirlenmiş herhangi bir performans kriterini karşılayıp karşılamadığını sağlamaktır. Bu şunları içerir:

- Makine Öğrenimi işlevsel performans kriterleri (bkz. Bölümler 5.1 ve 5.2)
- Yalnızca modele uygun olan MÖ modelinin işlevsel olmayan kabul kriterleri, örneğin eğitim hızı, tahmin hızı, kullanılan hesaplama kaynakları, uyarlanabilirlik ve şeffaflık.

Makine Öğrenimi modeli testi ayrıca, MÖ çerçevesi, algoritma, model, model ayarları ve hiperparametrelerin seçiminin mümkün olan en optimal seviyede olup olmadığını belirlemeyi amaçlar. Uygun olduğunda, MÖ model testi, beyaz kutu kapsama kriterlerini karşılamak için test etmeyi de içerebilir (bkz. Bölüm 6.2). Seçilen model daha sonra diğer bileşenlerle, Yapay Zekâ ve Yapay Zekâ olmayan bileşenlerle entegre edilir.

7.2.3 Bileşen Testi

Bileşen testi, kullanıcı arayüzleri ve iletişim bileşenleri gibi model olmayan bileşenler için geçerli olan geleneksel bir test seviyesidir.

7.2.4 Bileşen Entegrasyon Testi

Bileşen entegrasyon testi, sistem bileşenlerinin (hem Yapay Zekâ hem de Yapay Zekâ olmayan) doğru şekilde etkileşime girdiğinden emin olmak için gerçekleştirilen geleneksel bir test seviyesidir. Bu test, veri hattından gelen girdilerin model tarafından beklenildiği gibi alınıp alınmadığını ve model tarafından üretilen tahminlerin ilgili sistem bileşenleriyle (örneğin, kullanıcı arayüzü) değiştirilip onlar tarafından doğru şekilde kullanılıp kullanılmadığını test eder. YZ bir hizmet olarak sağlanıyorsa (bkz. Bölüm 1.7), bileşen entegrasyon testinin bir parçası olarak sağlanan hizmetin API testinin yapılması normaldir.

7.2.5 Sistem Testi

Sistem testi, entegre bileşenlerin (hem Yapay Zekâ hem de Yapay Zekâ olmayan) tam sistemini, hem işlevsel hem de işlevsel olmayan açılardan, beklenildiği gibi performans gösterdiğinden emin olmak için gerçekleştirilen geleneksel bir test seviyesidir. Bu, işletimsel ortamı yakından temsil eden bir test ortamında gerçekleştirilir. Sisteme bağlı olarak, bu test, beklenen işletim ortamında alan denemeleri şeklinde veya bir simülâtörde (örneğin, test senaryoları işletim ortamında tehlikeli veya zor kopyalanabilirse) gerçekleştirilebilir.

Sistem testi sırasında, MÖ işlevsel performans kriterleri, modelin tam bir sistem içinde yerleştirildiğinde başlangıçtaki MÖ model testlerinden elde edilen test sonuçlarının olumsuz

etkilenmediğinden emin olmak için yeniden test edilir. Bu test, Yapay Zekâ bileşeninin bilinçli olarak değiştirildiği durumlarda (örneğin, bir DNN'yi (Derin Sinir Ağı) boyutunu azaltmak için sıkıştırma) özellikle önemlidir.

Sistem testi ayrıca, sistemin birçok işlevsel olmayan gereksinimini test ettiğimiz test seviyesidir. Örneğin, dirençlilik testi yapılabilir veya sistemin açıklanabilirliği test edilebilir. Uygun olduğunda, donanım bileşenleriyle (örneğin, sensörler) arayüzler, sistem testinin bir parçası olarak test edilebilir.

7.2.6 Kabul Testi

Kabul testi, tüm sistemin müşteri tarafından kabul edilebilir olup olmadığını belirlemek için kullanılan geleneksel bir test seviyesidir. YZ tabanlı sistemler için kabul kriterlerinin tanımlanması zor olabilir (bkz. Bölüm 8.8). YZ bir hizmet olarak sağlanıyorsa (bkz. Bölüm 1.7), kabul testi, hizmetin amaçlanan sistem için uygunluğunu belirlemek ve örneğin, MÖ işlevsel performans kriterlerinin yeterince karşılanıp karşılanmadığını belirlemek için gerekebilir.

7.3 YZ Tabanlı Sistemlerin Test Edilmesi İçin Test Verileri

Test edilecek duruma ve test edilen sisteme (SUT) bağlı olarak, test verilerinin elde edilmesi bazı zorluklarla karşılaşabilir. YZ tabanlı sistemler için test verileriyle başa çıkma ile ilgili potansiyel bazı zorluklar şunları içerebilir:

- Büyük veri (yüksek hacimli, yüksek hızlı ve yüksek çeşitlilikte veri) oluşturulması ve yönetilmesi zor olabilir. Örneğin, yüksek hızda büyük miktarda görüntü ve ses tüketen bir sistem için temsilci test verileri oluşturmak zor olabilir.
- Girdi verisi zamanla değişebilir, özellikle gerçek dünyadaki olayları temsil ediyorsa. Örneğin, bir yüz tanıma sistemi için test etmek için kaydedilen fotoğraflar, gerçek hayatta birkaç yıl boyunca insanların yaşlanmasını temsil etmek için "yaşlandırılması" gerekebilir.
- Kişisel veya başka türlü gizli verilerin temizlenmesi, şifrelenmesi veya sansürlenmesi için özel tekniklere ihtiyaç duyulabilir. Kullanım için yasal onay da gerekebilir.
- Test uzmanları veri temini ve veri ön işleme için veri bilimcileri ile aynı uygulamayı kullandıklarında, bu adımlardaki hatalar gizlenebilir.

7.4 YZ Tabanlı Sistemlerde Otomasyon Yanlılığının Test Edilmesi

Yapay Zekâ tabanlı sistemlerin bir kategorisi, insanlara karar verme sürecinde yardımcı olur. Ancak, zaman zaman insanların bu sistemlere fazla güvenme eğilimi gösterdiği durumlar olur. Bu yanlış yerleştirilmiş güven, otomasyon yanlılığı veya rehabet yanlılığı olarak adlandırılabilir ve iki biçimde ortaya çıkar.

- Otomasyon/güven yanlılığının ilk şekli, insanın sistem tarafından sağlanan önerileri kabul etmesi ve diğer kaynaklardan gelen girdileri (kendileri de dahil olmak üzere) göz ardı etmesidir. Örneğin, bir insanın verileri bir forma girmesi gereken bir prosedür, Makine Öğrenimini kullanarak formu otomatik olarak doldurarak iyileştirilebilir ve insan daha sonra bu verileri doğrular. Bu tür otomasyon yanlılığının genellikle kararların kalitesini %5 oranında düşürdüğü gösterilmiştir, ancak bu oran sistem bağlamına bağlı olarak çok daha yüksek olabilir. Benzer şekilde, yazılan metnin (örneğin, cep telefonu mesajlarında) otomatik düzeltilmesi genellikle hatalıdır ve anlamı değiştirebilir. Kullanıcılar genellikle bunu fark etmez ve hatayı düzeltmezler.
- Otomasyon/güven yanlılığının ikinci şekli, insanın, sistemin yetersiz bir şekilde izlenmesi nedeniyle bir

sistem hatasını kaçırmasıdır. Örneğin, yarı otonom araçlar giderek daha fazla kendi kendine sürmeye başlıyor, ancak hala yaklaşan bir kazada insanın devralması gerekiyor. Tipik olarak, insan araç yolcusu aracın kontrolünü sağlama yeteneğine olan güvenini zamanla artırır ve daha az dikkat etmeye başlar. Bu, gerektiğinde uygun şekilde tepki veremedikleri bir duruma yol açabilir.

Her iki senaryoda da test uzmanlarının insanların karar verme sürecinin nasıl etkilenebileceğini anlaması ve sistem önerilerinin kalitesini ve temsilci kullanıcılar tarafından sağlanan karşılık gelen insan girdisinin kalitesini test etmesi gerekmektedir.

7.5 YZ Bileşeninin Belgelendirilmesi

Bir Yapay Zekâ bileşeninin belgelendirilmesi tipik olarak şunları içerir:

- Genel: Tanımlayıcılar, açıklama, geliştirici bilgileri, donanım gereksinimleri, lisans ayrıntıları, sürüm, tarih ve iletişim noktası.
- Tasarım: Varsayımlar ve teknik kararlar.
- Kullanım: Birincil ve ikincil kullanım durumları, tipik kullanıcılar, öz-ders alma yaklaşımı, bilinen yanlılık, etik konular, güvenlik konuları, şeffaflık, karar eşikleri, platform ve kavram kayması.
- Veri Setleri: Özellikler, toplama, kullanılabilirlik, ön işleme gereksinimleri, kullanım, içerik, etiketleme, boyut, gizlilik, güvenlik, yanlılık/dürüstlük ve kısıtlamalar.
- Test: Test veri seti (açıklama ve kullanılabilirlik), test bağımsızlığı, test sonuçları, sağlamlık, açıklanabilirlik, kavram kayması ve taşınabilirlik için test yaklaşımı.
- Eğitim ve Makine Öğrenimi İşlevsel Performansı: Makine Öğrenimi algoritması, ağırlıklar, doğrulama veri seti, Makine Öğrenimi işlevsel performans metriklerinin seçimi, Makine Öğrenimi işlevsel performans metrikleri için eşikler ve gerçek MÖ işlevsel performans metrikleri.

Net belgelendirme, YZ tabanlı sistemin uygulanması hakkında şeffaflık sağlayarak testi iyileştirmeye yardımcı olur. Test için önemli olan belgelendirme alanları şunlardır:

- Sistemin amacı ve işlevsel ve işlevsel olmayan gereksinimlerin belirlenmesi. Bu tür belgeler genellikle testin temelini oluşturur.
- Farklı Yapay Zekâ ve Yapay Zekâ dışı bileşenlerin nasıl etkileşimde bulunduğunu açıklayan mimari ve tasarım bilgileri. Bu, entegrasyon testi hedeflerinin belirlenmesine destek olur ve sistem yapısının beyaz kutu testi için bir temel sağlayabilir.
- İşletim ortamının belirlenmesi. Bu, sistemin otonomisini, esnekliğini ve uyarlanabilirliğini test ederken gereklidir.
- İlişkili meta veriler dahil olmak üzere tüm girdilerin kaynağı. Bu, aşağıdaki yönlerin test edilmesi gerektiğinde net bir şekilde anlaşılmalıdır:
 - Güvenilir olmayan girdilerin işlevsel doğruluğu
 - Açık veya örtük örneklem yanlılığı
 - Esneklik, kendi kendine öğrenen sistemler için kötü veri girdilerinden yanlış öğrenmeyi de içerir
- Sistemin işletim ortamındaki değişikliklere uyum sağlamanın beklendiği yöntem. Bu, uyarlanabilirlik testi yapılırken test temeli olarak gereklidir.
- Beklenen sistem kullanıcılarının detayları. Bu, testin temsilî olarak yapılabilmesini sağlamak için gereklidir.

7.6 Kavram Kayması İçin Test Etme

İşletimsel çevre zamanla değişebilir ancak eğitilmiş model aynı oranda değişmeyebilir. Bu fenomen, kavram kayması olarak bilinir ve genellikle modelin çıktılarının giderek daha az doğru ve daha az kullanışlı hale gelmesine neden olur. Örneğin, pazarlama kampanyalarının etkisi zamanla potansiyel müşterilerin davranışlarında bir değişime neden olabilir. Bu tür değişiklikler mevsimsel olabilir veya sistem dışındaki kültürel, ahlaki veya toplumsal değişikliklerden kaynaklanan ani değişiklikler olabilir. Bu tür bir ani değişikliğin bir örneği olarak, COVID-19 salgınının etkisi ve satış projeksiyonları ve borsa için kullanılan modellerin doğruluğu üzerindeki etkisi verilebilir.

Kavram kaymasına yatkın olabilecek sistemler, üzerinde mutabık kalınan MÖ işlevsel performans kriterlerine göre düzenli olarak test edilmeli, böylece kavram kaymasının olası belirtileri mümkün olan en kısa sürede tespit edilir ve sorunun hafifletilmesi sağlanır. Tipik hafifletme yöntemleri arasında sistemi kullanımdan kaldırma veya sistemi yeniden eğitime bulunabilir. Yeniden eğitime durumunda, bu, güncel eğitim verileriyle gerçekleştirilecek ve ardından onay testi, regresyon testi ve muhtemelen güncellenmiş B-sisteminin orijinal A-sisteminden daha iyi performans göstermesi gereken bir tür A/B testi (bkz. Bölüm 9.4) yapılacaktır.

7.7 MÖ Sistemi için Bir Test Yaklaşımı Seçme

Bir Yapay Zekâ tabanlı sistem genellikle Yapay Zekâ ve yapay olmayan bileşenleri içerecektir. Bu tür bir sistem için test yaklaşımı, bir risk analizine dayanır ve hem geleneksel testleri hem de Yapay Zekâ bileşenleri ve Yapay Zekâ tabanlı sistemlere özgü faktörleri ele alan daha özelleşmiş testleri içerir.

Aşağıdaki liste, MÖ sistemlerine özgü bazı tipik riskleri ve bunlara karşılık gelen hafifletme önlemlerini sunar. Bu listenin yalnızca sınırlı sayıda örnek sunduğunu ve Makine Öğrenimi sistemlerine özgü, test yoluyla azaltılması gereken çok daha fazla risk bulunduğunu unutmayın.

Risk Unsurları	Açıklama ve Olası Hafifletme Önlemleri
Veri kalitesi beklenenden düşük olabilir.	Bu risk, farklı şekillerde sorun haline gelebilir ve her biri farklı yöntemlerle önlenabilir (bkz. Bölüm 4.4). Yaygın azaltma yöntemleri arasında incelemelerin, EDA'nın ve dinamik testlerin kullanılması bulunur.
İşletimsel veri hattı hatalı olabilir.	Bu risk, bireysel veri hattı bileşenlerinin dinamik test edilmesi ve tam veri hattının entegrasyon test edilmesiyle kısmen hafifletilebilir.
Modeli geliştirmek için kullanılan MÖ iş akışı, alt-optimal olabilir. (Bkz. Bölüm 3.2)	Bu risk aşağıdakilerden kaynaklanabilir: <ul style="list-style-type: none">• İzlenecek MÖ iş akışı konusunda önceden anlaşma eksikliği Zayıf bir iş akışı seçimi <ul style="list-style-type: none">• Veri mühendislerinin iş akışını izlememesi Uzmanlarla yapılan incelemeler, yanlış iş akışının seçilme olasılığını azaltabilirken, daha fazla pratik yönetim veya denetimler, iş akışının belirlenmesi ve uygulanması konularındaki anlaşmazlıkları ele alabilir.

Risk Unsurları	Açıklama ve olası Azaltmalar
MÖ çerçevesi, algoritma, model, model ayarları ve/veya hiperparametrelerin seçimi alt-optimal olabilir.	Bu risk, karar vericilerin uzmanlık eksikliği veya MÖ iş akışının değerlendirme ve uygulama adımlarının (veya test adımının) kötü uygulanmasından kaynaklanabilir. Uzmanlarla yapılan incelemeler, yanlış kararların alınma olasılığını azaltabilirken, daha iyi yönetim, iş akışının değerlendirme ve ayarlama (ve test) adımlarının izlenmesini sağlayabilir.
İstenen MÖ işlevsel performans kriterleri, MÖ bileşeninin bu kriterleri izole olarak karşılamasına rağmen, operasyonel olarak sunulmayabilir.	Bu risk, modelin eğitimi ve testi için kullanılan veri setlerinin operasyonel olarak karşılaşılan veriyle temsilci olmamasından kaynaklanabilir. Uzmanlar (veya kullanıcılar) tarafından seçilen veri setlerinin incelenmesi, seçilen verinin temsilci olmadığı olasılığını azaltabilir.
İstenen MÖ işlevsel performans kriterleri karşılanmış olabilir, ancak kullanıcılar sunulan sonuçlardan memnun olmayabilir.	Bu risk, yanlış performans kriterlerinin seçilmesine (örneğin, yüksek hassasiyet yerine yüksek hatırlama seçilmiş olabilir) bağlı olabilir. Uzmanlarla yapılan incelemeler, yanlış MÖ işlevsel performans metriklerinin seçilme olasılığını azaltabilir veya deneyim temelli testler de uygun olmayan kriterleri belirleyebilir. Risk ayrıca kavram kaymasından kaynaklanabilir, bu durumda operasyonel sistemlerin daha sık test edilmesi riski azaltabilir.
İstenen MÖ işlevsel performans kriterleri karşılanmış olabilir, ancak kullanıcılar sunulan hizmetten memnun olmayabilir.	Bu risk, sistemin işlevsel olmayan gereksinimlerine yeterince odaklanılmamasının bir sonucu olabilir. YZ tabanlı sistemler için kalite özelliklerinin yelpazesi, ISO/IEC 25010'da listelenenlerden daha geniştir (bkz. Bölüm 2). Kalite özelliklerini önceliklendirmek ve ilgili işlevsel olmayan testleri gerçekleştirmek için risk odaklı bir yaklaşım kullanılabilir. Alternatif olarak, sorun deneyime dayalı testlerle, sistem testinin bir parçası olarak tanımlanabilecek faktörlerin bir kombinasyonundan kaynaklanıyor olabilir. Bölüm 8, bu özelliklerin nasıl test edileceği konusunda rehberlik sağlar.

Risk Unsurları	Açıklama ve olası Azaltmalar
Kendi kendine öğrenen sistem kullanıcıların beklediği hizmeti sağlayamıyor olabilir.	<p>Bu risk çeşitli nedenlere bağlı olabilir, örneğin:</p> <ul style="list-style-type: none">• Sistem tarafından kullanılan verilerin kendi kendine öğrenme için uygun olmayabileceği durumlar olabilir. Bu durumda, uzmanların incelemeleri sorunlu verileri tespit edebilir.• Yeni kendi kendine öğrenilen işlevsellik kabul edilemez olduğunda sistem başarısız olabilir. Bu, önceki işlevsellikle performans karşılaştırması da içeren otomatik regresyon testleriyle hafifletilebilir.• Kullanıcıların beklemediği bir şekilde öğrenme gerçekleşiyor olabilir, ki bu deneyime dayalı testlerle tespit edilebilir.
Kullanıcılar sistemin kararlarını nasıl belirlediğini anlamadıkları için hayal kırıklığına uğrayabilirler	<p>Bu risk açıklanabilirlik, yorumlanabilirlik ve/veya şeffaflık eksikliğinden kaynaklanıyor olabilir. Bu özelliklerin test edilmesiyle ilgili ayrıntılar için Bölüm 8'e bakın.</p>
Kullanıcılar, veriler eğitim verilerine benzer olduğunda modelin mükemmel tahminler sağladığını ancak aksi durumda kötü sonuçlar verdiğini görebilir.	<p>Bu risk, modelin eğitim veri kümesinden tamamen bağımsız bir veri kümesiyle test edilmesiyle veya deneyime dayalı testlerin gerçekleştirilmesiyle tespit edilebilecek aşırı uyumdan (bkz. Bölüm 3.5.1) kaynaklanıyor olabilir.</p>

8. Yapay Zekâ'ya Özgü Kalite Özelliklerini Test Etme – 150 Dakika

Anahtar Kelimeler

Test Doğrulayıcı

YZ-Spesifik Anahtar Kelimeleri

Algoritmik Yanlılık, Otonom Sistem, Otonomi, Uzman Sistem, Açıklanabilirlik, Uygunsuz Yanlılık, Yorumlanabilirlik, LIME Metodu, Makine Öğrenimi Eğitim Verisi, Belirsiz Sistem, Olasılıksal Sistem, Örneklem Yanlılık, Kendini Öğreten Sistem, Şeffaflık

Bölüm 8- Öğrenme Amaçları:

8.1 Kendi Kendine Öğrenen Sistemleri Test Etmenin Zorlukları

YZ-8.1.1 K2 Yapay Zekâ tabanlı sistemlerin kendi kendine öğrenmenin yarattığı test zorluklarını açıklayın.

8.2 Otonom Yapay Zekâ Tabanlı Sistemlerin Test Edilmesi

YZ-8.2.1 K2 Otonom Yapay Zekâ tabanlı sistemlerin nasıl test edildiğini tarif edin.

8.3 Algoritmik, Örnek ve Uygunsuz Yanlılık Testi

YZ-8.3.1 K2 Bir Yapay Zekâ tabanlı sistemdeki yanlılığın nasıl test edileceğini açıklayın.

8.4 Olasılıksal ve Deterministik Olmayan Yapay Zekâ Tabanlı Sistemlerin Test Edilmesinde Karşılaşılan Zorluklar

YZ-8.4.1 K2 Olasılıksal ve belirsiz doğanın yarattığı test zorluklarını açıklayın.

8.5 Karmaşık Yapay Zekâ Tabanlı Sistemlerin Test Edilmesindeki Zorluklar

YZ-8.5.1 K2 Yapay Zekâ tabanlı sistemlerin karmaşıklığı tarafından yaratılan test zorluklarını açıklayın.

8.6 Yapay Zekâ Tabanlı Sistemlerin Şeffaflığının, Yorumlanabilirliğinin ve Açıklanabilirliğinin Test Edilmesi

YZ-8.6.1 K2 Yapay Zekâ tabanlı sistemlerin şeffaflığı, yorumlanabilirliği ve açıklanabilirliğinin nasıl test edileceğini tarif edin.

HO-8.6.1 H2 Test uzmanlarının açıklanabilirliği nasıl kullanabileceğini göstermek için bir araç kullanın.

8.7 Yapay Zekâ Tabanlı Sistemler için Doğrulayıcıların Test Edilmesi

YZ-8.7.1 K2 Yapay Zekâ tabanlı sistemlerin özgül özelliklerinden kaynaklanan test doğrulayıcılarının oluşturulmasındaki zorlukları açıklayın.

8.8 Test Amaçları ve Kabul Kriterleri

YZ-8.8.1 K4 Verilen bir Yapay Zekâ tabanlı sistemin Yapay Zekâ'ya özgü kalite özellikleri için uygun test amaçları ve kabul kriterlerini seçin.

8.1 Kendi Kendine Öğrenen Sistemleri Test Etmenin Zorlukları

Kendi kendine öğrenme sistemleri test etmek bazı potansiyel zorluklar içerebilir (bu sistemler hakkında daha fazla detay için Bölüm 2'ye bakınız), şunlar dahil:

- Beklenmedik değişiklik: Sistemin çalışması gereken orijinal gereksinimler ve kısıtlamalar genellikle bilinir; ancak sistem tarafından yapılan değişiklikler hakkında az bilgi bulunabilir veya hiç bulunmayabilir. Orijinal gereksinimler ve tasarıma karşı (ve belirtilen herhangi bir kısıtlamaya) test yapmak genellikle mümkündür, ancak sistem yenilikçi bir uygulama geliştirmişse veya bir çözümü oyunlaştırmışsa (bu uygulamanın ne olduğu görülemezse), bu yeni uygulama için uygun testlerin tasarlanması zor olabilir. Ayrıca, sistemler kendilerini (ve çıktılarını) değiştirdiğinde, daha önce geçen testlerin sonuçları değişebilir. Bu, bir test tasarımı zorluğudur. Bu, sistemin davranışını değiştirdiğinde uygun testler tasarlayarak ve böylece potansiyel bir regresyon testi sorununu önleyerek ele alınabilir. Ancak, gözlemlenen yeni sistem davranışlarına dayanarak yeni testlerin tasarlanması da gerekebilir.
- Karmaşık kabul kriterleri: Sistem kendi kendine öğrendiğinde iyileşme beklentilerinin tanımlanması gerekebilir. Örneğin, sistem kendini değiştirirse, genel işlevsel performansının iyileşmesi beklenir. Ayrıca, basit bir "iyileşme" dışında herhangi bir şey belirtmek hızla karmaşık hale gelebilir. Örneğin, minimum bir iyileşme bekleniyor olabilir (sadece herhangi bir iyileşme değil), veya gerekli iyileşme çevresel faktörlere bağlı olabilir (örneğin, çevresel faktör F, Y'den fazla değişirse, işlevsellik X'te minimum %10 iyileşme gereklidir). Bu sorunlar, daha karmaşık kabul kriterlerine karşı belirtmek ve test yaparak ve mevcut sistem temel işlevsel performansının sürekli bir kaydını tutarak ele alınabilir.
- Yetersiz test süresi: Sistemin, farklı senaryolar göz önünde bulundurulduğunda ne kadar hızlı öğrenip adapte olması gerektiğini bilmek gerekebilir. Bu kabul kriterlerini belirlemek ve elde etmek zor olabilir. Eğer sistem hızlı bir şekilde adapte olursa, her değişiklikten sonra yeni testleri manuel olarak yürütmek için yeterli zaman olmayabilir, bu nedenle sistemin kendini değiştirdiğinde otomatik olarak çalıştırılacak testler yazmak gerekebilir. Bu zorluklar, uygun kabul kriterlerinin belirlenmesi (bkz. Bölüm 8.8) ve otomatik sürekli testlerle aşılabılır.
- Kaynak gereksinimleri: Sistem gereksinimleri, sistem kendini öğrenirken veya adapte ederken kullanılmasına izin verilen kaynaklar için kabul kriterlerini içerebilir. Bu, örneğin, iyileştirmek için kullanılacak işlem süresi ve bellek miktarını içerebilir. Ayrıca, bu kaynak kullanımının işlevsellik veya doğrulukta ölçülebilir bir iyileşme ile bağlantılı olup olmaması gerektiği üzerinde düşünülmesi gerekir. Bu zorluk, kabul kriterlerinin belirlenmesini etkiler.
- İşletim ortamının yetersiz belirtilmesi: Kendi kendine öğrenen bir sistem, aldığı çevresel girdiler beklenen aralıkların dışındaysa veya eğitim verilerinde yansıtılmamışsa değişebilir. Bu girdiler, veri zehirlenme şeklindeki saldırılar olabilir (Bölüm 9.1.2'ye bakınız). İşletim ortamlarının ve çevresel değişikliklerin tam aralığını tahmin etmek ve bu nedenle temsili test senaryolarının ve çevresel gereksinimlerin tam setini belirlemek zor olabilir. İdeal olarak, sistemin cevap vermesi beklenen işletim ortamındaki olası değişikliklerin tam kapsamı, kabul kriterleri olarak tanımlanacaktır.
- Karmaşık test ortamı: Test ortamını yönetmek, tüm potansiyel yüksek riskli işletim ortamı değişikliklerini taklit edebilmesini sağlamak bir zorluktur ve test araçlarının kullanımını (örneğin, bir hata enjeksiyon aracı) gerektirebilir. İşletim ortamının doğasına bağlı olarak, bu girdileri ve sensörleri manipüle ederek veya sistem test edilebilecek farklı fiziksel ortamlara erişim sağlayarak test edilebilir.
- İstenmeyen davranış değişiklikleri: Kendi kendine öğrenen bir sistem, girdilerine dayanarak davranışını değiştirir ve test uzmanlarının bunun olmasını engellemesi mümkün olmayabilir. Bu, örneğin, üçüncü parti bir sistem kullanılıyorsa veya üretim sistemi test ediliyorsa ortaya çıkabilir.

Aynı testler tekrarlandığında, kendi kendine öğrenen bir sistem bu testlere yanıt vermekte daha etkili hale gelebilir, bu da sistemin uzun vadeli davranışını etkileyebilir. Bu nedenle, testlerin kendi kendine öğrenen bir sistemin davranışını olumsuz yönde değiştirmesine neden olan bir durumu önlemek önemlidir. Bu, test senaryosu tasarımı ve test yönetimi için bir zorluktur.

8.2 Otonom Yapay Zekâ Sistemlerin Test Edilmesi

Otonom sistemlerin, ne zaman insan müdahalesine ihtiyaç duyduklarını ve ne zaman duymadıklarını belirleyebilmeleri gerekmektedir. Bu nedenle, YZ tabanlı sistemlerin otonomisini test etmek, sistemin bu karar verme yeteneğini kullanabileceği koşulların oluşturulmasını gerektirir.

Otonomi için test yapmak şunları gerektirebilir:

- Sistemin, kontrolü bırakması gerektiği belirli bir senaryo için insan müdahalesi talep edip etmediğini test etme. Bu tür senaryolar, işletim ortamında bir değişikliği veya sistemin otonomisinin sınırlarını aşmasını içerebilir.
- Sistemin, belirlenen bir süre sonrasında kontrolü bırakması gerektiğinde insan müdahalesi talep edip etmediğini test etme.
- Sistemin, hala otonom olarak çalışması gerektiğinde gereksiz yere insan müdahalesi talep edip etmediğini test etme.

İşletim ortamına uygulanan sınır değer analizini kullanmak, bu test için gerekli koşulları üretmek açısından yardımcı olabilir. Otonomiye belirleyen parametrelerin işletim ortamında nasıl kendini gösterdiğini tanımlamak ve otonominin doğasına bağlı olarak test senaryolarını oluşturmak zorlayıcı olabilir.

8.3 Algoritmik, Örneklem ve Uygunuz Yanlılık Testi

Bir Makine Öğrenimi sistemi, farklı yanlılıklara ve uygunuz yanlılığın giderilmesi için yapılan eylemlere karşı değerlendirilmelidir. Bu, uygunuz yanlılığı dengelemek için kasıtlı olarak pozitif yanlılık getirilmesini içerebilir.

Bağımsız bir veri seti ile test yapmak sık sık yanlılığı tespit edebilir. Ancak, Makine Öğrenimi algoritmasının istenmeyen yanlılık yaratmak için görünüşte ilgisiz özelliklerin kombinasyonlarını kullanabilmesi nedeniyle, yanlılığa neden olan tüm verileri tanımlamak zor olabilir.

Yapay Zekâ tabanlı sistemler algoritmik yanlılık, örneklem yanlılık ve uygunuz yanlılık için test edilmelidir (Bölüm 2.4'e bakınız). Bu, şunları içerebilir:

- Algoritmik yanlılığın olup olmadığını belirlemek için, modelin eğitimi, değerlendirilmesi ve ayar aktiviteleri sırasında analiz yapmak,
- Örneklem yanlılığın varlığı tespit edilmek için, eğitim verisinin kaynağını ve onu elde etmek için kullanılan süreçleri gözden geçirmek
- Verinin örneklem yanlılığa yol açabilecek bir şekilde etkilenip etkilenmediğini belirlemek için, verinin Makine Öğrenimi iş akışının bir parçası olarak ön işlemlerini gözden geçirmek
- Sistem girdilerindeki değişikliklerin, çok sayıda etkileşim üzerinden sistem çıktılarını nasıl etkilediğinin ölçülmesi ve sonuçların, sistemin uygun olmayan bir şekilde yanlı olabileceği veya aleyhine olabileceği insanlar veya nesnelerin gruplarına göre incelenmesi. Bu, 8.6'da tartışılan LIME (Yerel Yorumlanabilir Model-Agnostik Açıklamalar) yöntemine benzer ve üretim ortamında ve aynı zamanda yayına hazırlık aşamasındaki testlerin bir parçası olarak gerçekleştirilebilir.
- Yanlılık ile potansiyel olarak ilgili olabilecek girdi verilerinin özellikleri hakkında ek bilgilerin elde edilmesi ve bunun sonuçlarla ilişkilendirilmesi.

Bu, örneğin, demografik verilerle ilgili olabilir, bu da insan gruplarını etkileyen uygunsuz yanlılığı test ederken uygun olabilir, burada bir grubun üyeliği yanlılığı değerlendirirken önemlidir ancak modelin bir girdisi değildir. Bu, yanlılığın, girdi verilerinde açıkça mevcut olmayan ancak algoritma tarafından çıkarılan "gizli" değişkenlere dayanabileceği anlamına gelir.

8.4 Olasılıksal ve Deterministik Olmayan YZ Tabanlı Sistemlerin Test Edilmesinde Karşılaşılan Zorluklar

Çoğu olasılıksal sistem aynı zamanda belirsizdir ve bu nedenle aşağıdaki test zorlukları listesi tipik olarak bu özelliklere sahip Yapay Zekâ tabanlı sistemler için geçerlidir:

- Aynı ön koşullar ve girdilerle yapılan bir testten birden fazla, geçerli sonuç çıkabilir. Bu, beklenen sonuçların tanımını daha zor hale getirebilir ve şu durumlarda zorluklara neden olabilir:
 - testler onay testi için yeniden kullanıldığında.
 - testler regresyon testi için yeniden kullanıldığında.
 - testlerin tekrarlanabilirliği önemli olduğunda.
 - testler otomatikleştirildiğinde.
- Test uzmanı tipik olarak, testin geçip geçmediğine dair makul kontroller yapabilmek için gerekli sistem davranışı hakkında daha derin bir bilgiye ihtiyaç duyar; sadece beklenen test sonucu için kesin bir değer belirtmek yerine. Örneğin, test uzmanları, geleneksel sistemlere kıyasla daha sofistike beklenen sonuçlar tanımlamak zorunda kalabilir. Bu beklenen test sonuçları toleransları içerebilir (örneğin, "gerçek sonuç, optimal çözümün %2 içinde mi?").
- Sistemin olasılıksal doğası nedeniyle bir testten tek bir kesin çıktı alınamadığında, test uzmanının istatistiksel olarak geçerli bir test sonucu üretmek için bir testi birkaç kez çalıştırması sıklıkla gereklidir.

8.5 Karmaşık YZ Tabanlı Sistemlerin Test Edilmesindeki Zorluklar

Yapay Zekâ tabanlı sistemler sıklıkla insanların gerçekleştiremeyeceği kadar karmaşık görevlerin uygulanması için kullanılır. Bu, test uzmanlarının beklenen sonuçları normalde yapacakları gibi belirleyememeleri nedeniyle bir test doğrulayıcı problemine yol açabilir (Bölüm 8.7'ye bakınız). Örneğin, Yapay Zekâ tabanlı sistemler genellikle büyük veri hacimlerinde desenlerin tanımlanması için kullanılır. Bu tür sistemler, insanların, çok analiz yapsalar bile, basitçe manuel olarak bulamayacakları desenleri bulabilmeleri nedeniyle kullanılır. Bu tür sistemlerin gereken davranışını yeterli derinlikte anlamak ve beklenen sonuçları üretebilmek zor olabilir.

Benzer bir problem, bir Yapay Zekâ tabanlı sistemin iç yapısının yazılım tarafından üretilmesi ve insanların anlayamayacağı kadar karmaşık hale gelmesi durumunda ortaya çıkar. Bu, Yapay Zekâ tabanlı sistemin sadece bir kara kutu olarak test edilebileceği duruma yol açar. İç yapısı görünür olsa bile, bu test ile yardımcı olacak ek kullanışlı bilgiler sağlamaz.

Yapay Zekâ tabanlı sistemlerin karmaşıklığı, olasılıksal sonuçlar sunduklarında ve doğaları gereği belirsiz olduklarında artar (Bölüm 8.4'e bakınız).

Belirsiz sistemlerle ilgili sorunlar, birkaç etkileşimli bileşenden oluşan bir Yapay Zekâ tabanlı sistem söz konusu olduğunda daha da kötüleşir. Örneğin, bir yüz tanıma sistemi, bir resim içindeki yüzü tanımlamak için bir model ve tanımlanan yüzün hangisi olduğunu tanımak için ikinci bir model kullanabilir. Yapay Zekâ bileşenleri arasındaki etkileşimler karmaşık ve anlaşılması zor olabilir, bu da tüm riskleri tanımlamayı ve sistemi yeterince doğrulayan testler tasarlamayı zorlaştırır.

8.6 YZ Tabanlı Sistemlerin Şeffaflığının, Yorumlanabilirliğinin ve Açıklanabilirliğinin Test Edilmesi

Sistem geliştiricileri tarafından sistem nasıl uygulandığına dair bilgiler sağlanabilir. Bu, eğitim verilerinin kaynakları, etiketlenmenin nasıl yapıldığı ve sistem bileşenlerinin nasıl tasarlandığı gibi konuları içerebilir. Bu bilgiler mevcut değilse, testlerin tasarımını zorlaştırabilir. Örneğin, eğitim verisi bilgisi mevcut değilse, bu verideki potansiyel boşlukları tanımlamak ve bu boşlukların etkisini test etmek zorlaşır. Bu durum, kara kutu ve beyaz kutu testiyle karşılaştırılabilir ve benzer avantajlar ve dezavantajlar sunar.

Şeffaflık, veri ve algoritma hakkında belirlenen bilgilerin gerçek uygulama ile karşılaştırılması ve ne kadar yakından eşleştiklerinin belirlenmesiyle test edilebilir.

Makine Öğrenimi ile, spesifik bir girdi ile spesifik bir çıktı arasındaki bağlantıyı açıklamak, geleneksel sistemlerle karşılaştırıldığında genellikle daha zor olabilir. Bu düşük düzeyde açıklanabilirlik, çıktıyı üreten modelin kendisinin kod (algoritma) tarafından üretilmesi ve insanların bir problem hakkında düşündüğü şekli yansıtmaması nedeniyledir. Farklı Makine Öğrenimi modelleri, farklı düzeylerde açıklanabilirlik sunar ve sistem için gereken gereksinimlere bağlı olarak seçilmelidir; bu gereksinimler açıklanabilirlik ve test edilebilirliği içerebilir.

Açıklanabilirliği anlamamanın bir yolu, test verilerine bozulmalar uygulanırken Makine Öğrenimi modelinin dinamik testi aracılığıyla. Bu şekilde açıklanabilirliği nicel olarak ölçmek ve bunun görsel açıklamalarını sağlamak için yöntemler mevcuttur. Bu yöntemlerden bazıları modelden bağımsızken, diğerleri belirli bir model türüne özgüdür ve buna erişim gerektirir. Keşifsel test kullanılarak da bir modelin girdileri ve çıktıları arasındaki ilişki daha iyi anlaşılabilir.

LIME metodu, modelden bağımsızdır ve dinamik olarak enjekte edilen girdi bozulmalarını ve çıktıların analizini kullanarak, test uzmanlarına girdiler ile çıktılar arasındaki ilişkiye dair bir görünüm sağlar. Bu, model açıklanabilirliği sağlamak için etkili bir yöntem olabilir. Ancak, bu yöntem sadece çıktılar için olası nedenler sunmakla sınırlıdır, kesin bir neden sunmaz ve tüm algoritmalar için uygun değildir.

Bir YZ tabanlı sistemin yorumlanabilirliği, bunun kimin için geçerli olduğuna büyük ölçüde bağlıdır. Farklı paydaşlar, altında yatan teknolojiyi ne kadar iyi anlamaları gerektiği konusunda farklı gereksinimlere sahip olabilir.

Hem yorumlanabilirlik hem de açıklanabilirlik için anlama düzeyini ölçmek ve test etmek, paydaşların yetenek seviyeleri farklılık gösterebileceği ve muhtemelen anlamayabileceği için zor olabilir. Ayrıca, birçok sistem türü için tipik paydaş profillerini belirlemek zor olabilir. Yapıldığı yerlerde, bu test genellikle kullanıcı anketleri ve/veya anketler şeklini alır.

8.6.1 Uygulamalı Egzersiz: Model Açıklanabilirliği

Daha önce oluşturulan modele dayanarak açıklanabilirlik sağlamak için uygun bir araç kullanın. Örneğin, bir görüntü sınıflandırma modeli veya bir metin sınıflandırma modeli için, LIME gibi modelden bağımsız bir yöntem uygun olabilir.

Öğrenciler, özellikle, girdilerdeki özelliklerin çıktıları nasıl etkilediği konusunda model kararlarının açıklamalarını üretmek için aracı kullanmalıdır.

8.7 Yapay Zekâ Tabanlı Sistemler için Testin Doğruluk Ölçütleri

Yapay Zekâ tabanlı sistemlerin test edilmesiyle ilgili önemli bir sorun, beklenen sonuçların belirlenmesi olabilir. Bir test doğrulayıcı, bir testin beklenen sonucunu belirlemek için kullanılan kaynaktır [I01]. Beklenen sonuçları belirlemedeki bir zorluk, test doğrulayıcı problemi olarak bilinir.

Karmaşık, belirsiz veya olasılıksal sistemlerde, YZ tabanlı sistemin tahmin etmeye çalıştığı gerçek dünyadaki asıl sonucu "asıl gerçek" (yani, YZ tabanlı sistemin tahmin etmeye çalıştığı gerçek dünyadaki gerçek sonuç) bilmeden bir test doğruluk ölçütü oluşturmak zor olabilir. Bu "asıl gerçek", bir test doğruluk ölçütünden farklıdır, çünkü bir test doğruluk ölçütü mutlaka bir beklenen değer sağlamayabilir, ancak sistemin doğru çalışıp çalışmadığını belirlemek için bir mekanizma sunar.

Yapay Zekâ tabanlı sistemler evrilebilir (Bölüm 2.3'e bakınız) ve kendi kendini öğrenen sistemlerin test edilmesi (Bölüm 8.1'e bakınız) de kendilerini değiştirdikleri için test doğruluk ölçütü problemlerinden muzdarip olabilir ve bu da sistemin işlevsel beklentilerini sık sık güncellemeyi gerektirebilir.

Etkili bir test doğruluk ölçütü elde etmede zorluğun başka bir nedeni, birçok durumda, yazılım davranışının doğruluğunun subjektif olmasıdır. Sanal asistanlar (örneğin, Siri ve Alexa), farklı kullanıcıların sıklıkla oldukça farklı beklentilere sahip olduğu ve kelime seçimleri ile konuşma açıklığına bağlı olarak farklı sonuçlar yaşayabileceği bu problem için bir örnektir.

Bazı durumlarda, beklenen sonuç sınırlar veya toleranslar ile tanımlanabilir. Örneğin, bir otonom aracın durma noktası, belirli bir noktanın maksimum mesafesi içinde olarak tanımlanabilir. Uzman sistemler bağlamında, beklenen sonuçların belirlenmesi, bir uzmana danışarak gerçekleştirilebilir (uzmanın görüşünün hala yanlış olabileceğini not almak gerekir). Bu tür durumlarda dikkate alınması gereken birkaç önemli faktör vardır:

- İnsan uzmanlar yetenek seviyeleri açısından farklılık gösterir. Sistemin yerine getirmesi amaçlanan uzmanlar kadar yetkin olacak şekilde ilgili uzmanlar dahil edilmelidir.
- Uzmanlar, aynı bilgi ile karşılaştıklarında birbirleriyle aynı fikirde olmayabilir.
- İnsan uzmanlar, kendi yargılarının otomasyonunu onaylamayabilir. Bu gibi durumlarda, potansiyel çıktılarının değerlendirilmesi çift taraflı kör olmalıdır (yani ne uzmanlar ne de çıktılarının değerlendiricileri hangi değerlendirmelerin otomatik yapıldığını bilmemelidir).
- İnsanlar, yanıtlarını genellikle (örneğin, "Emin değilim, ama..." gibi ifadelerle) çekince ile sunma eğilimindedir. Eğer bu tür bir çekince, Yapay Zekâ tabanlı sisteme mevcut değilse, yanıtları karşılaştırırken bu dikkate alınmalıdır.

Test doğruluk ölçütü problemi hafifletmeye yardımcı olabilecek test teknikleri mevcuttur, örneğin A/B testi (Bölüm 9.4'e bakınız), sırt sırta test (Bölüm 9.3'e bakınız) ve metamorfik test (Bölüm 9.5'e bakınız).

8.8 Test Amaçları ve Kabul Kriterleri

Bir sistemin test amaçları ve kabul kriterleri, algılanan ürün risklerine dayanmalıdır. Bu riskler sıklıkla gerekli kalite özelliklerinin analizinden tespit edilebilir. Bir Yapay Zekâ tabanlı sistem için kalite özellikleri, ISO/IEC 25010 [S06]'da (yani, fonksiyonel uygunluk, performans verimliliği, uyumluluk, kullanılabilirlik, güvenilirlik, güvenlik, bakım kolaylığı ve taşınabilirlik) geleneksel olarak düşünülenlerin yanı sıra aşağıdaki yönleri de içermelidir:

Bakış Açısı	Kabul Kriterleri
Uyarlanabilirlik	<ul style="list-style-type: none">Sistem, çevresindeki bir değişime adapte olduğunda doğru şekilde işlev görüp görmediğini ve işlevsel olmayan gereksinimleri karşılayıp karşılamadığını kontrol edin. Bu, otomatize edilmiş regresyon testi şeklinde uygulanabilir.Sistemin çevresindeki bir değişime adapte olmak için harcadığı süreyi kontrol edin.Sistem, çevresindeki bir değişime adapte olduğunda kullanılan kaynakları kontrol edin.
Esneklik	<ul style="list-style-type: none">Sistem, başlangıç spesifikasyonunun dışındaki bağlamlarda nasıl başa çıktığını değerlendirin. Bu, değişen işletim ortamında gerçekleştirilen otomatize edilmiş regresyon testi şeklinde uygulanabilir.Sistemin kendini yeni bir bağlama uyarlamak için harcadığı süreyi ve/veya kullanılan kaynakları kontrol edin.
Evrim	<ul style="list-style-type: none">Sistemin kendi deneyimlerinden ne kadar iyi öğrendiğini kontrol edin.Sistemin veri profilindeki değişikliklere (yani, kavram kayması) ne kadar iyi adapte olduğunu kontrol edin.
Otonomi	<ul style="list-style-type: none">Sistemin, tamamen otonom olması beklenen işletim zarfının dışına zorlandığında nasıl tepki verdiğini kontrol edin.Sistemin, tamamen otonom olması gerektiğinde insan müdahalesini talep etmeye "ikna edilip edilemeyeceğini" kontrol edin.
Şeffaflık, Yorumlanabilirlik ve Açıklanabilirlik	<ul style="list-style-type: none">Algoritma ve veri setine erişim kolaylığını gözden geçirerek şeffaflığı kontrol edin.Sistem kullanıcılarını sorgulayarak yorumlanabilirliği ve açıklanabilirliği kontrol edin ya da gerçek sistem kullanıcıları mevcut değilse, benzer bir arka plana sahip kişileri kullanın.
Uygunsuz Yanıllıktan Korunma	<ul style="list-style-type: none">Sistemlerin yanıllık tan etkilenebileceği durumlarda, bağımsız bir yanıllık tan arındırılmış test seti kullanarak veya uzman incelemecileri kullanarak bu test edilebilir.İstenmeyen yanıllıklar için harici verileri, örneğin nüfus sayımı verilerini kullanarak test sonuçlarını karşılaştırın (harici geçerlilik testi).
Etik	<ul style="list-style-type: none">Sistemi, Etik Kurallara Uygun Güvenilir Yapay Zekâ için EC Değerlendirme Listesi [R21] gibi uygun bir kontrol listesine karşı kontrol edin, bu da Güvenilir Yapay Zekâ (YZ) için Etik Kuralların ana gerekliliklerini destekler [R22].

Bakış Açısı	Kabul Kriteri
Olasılıksal Sistemler ve Belirsiz Sistemler	<ul style="list-style-type: none">Bu, kesin kabul kriterleri ile değerlendirilemez. Doğru çalıştığında, sistem aynı testler için biraz farklı sonuçlar döndürebilir.
Yan Etkiler	<ul style="list-style-type: none">Potansiyel olarak zararlı yan etkileri tanımlayın ve sistemin bu yan etkileri sergilemesine neden olacak testler üretmeye çalışın.
Ödül Hileciliği	<ul style="list-style-type: none">Bağımsız testler, bu testler test edilen zeki ajanın başarısını ölçme yönteminden farklı bir yöntem kullandığında ödül hileciliğini belirleyebilir.
Güvenlik	<ul style="list-style-type: none">Bu, dikkatlice değerlendirilmelidir, belki de bir sanal test ortamında (Bölüm 10.2'ye bakınız). Bu, bir sistemi kendine zarar vermesi için zorlama girişimlerini içerebilir.

Makine Öğrenimi sistemleri için, Makine Öğrenimi modeli için gereken Makine Öğrenimi işlevsel performans metrikleri belirtilmelidir (Bölüm 5'e bakınız).

9. YZ Tabanlı Sistemlerin Test Edilmesi için Yöntem ve Teknikler -245 dakika

Anahtar Kelimeler

A/B Testi, Karşıt Testler, Sırt Sırta Test Etme, Hata Tahminleme, Tecrübeye Dayalı Test, Keşif Testi, Metamorfik İlişki (MR), Metamorfik Test (MT), İkili Test, Sözde Sonucu Bilen, Sözde Test Sonucunu Bilen Problemi, Turlar

YZ-Spesifik Anahtar Kelimeleri

Karşıt Saldırısı, Karşıt Örneği, Veri Zehirlenmesi, Makine Öğrenim Sistemi, Eğitilmiş Model

Konu 9 Öğrenme Hedefleri:

9.1 Karşıt Saldırıları ve Veri Zehirlenmesi

YZ-9.1.1 (K2) Makine Öğrenimi sistemlerinin test edilmesinin karşıt saldırıları ve veri zehirlenmesini önlemeye nasıl yardımcı olabileceğini açıklar.

9.2 İkili Testler

YZ-9.2.1 (K2) Yapay Zekâ tabanlı sistemler için ikili testlerin nasıl kullanıldığını açıklar.

ÖÇ-9.2.1 (H2) Yapay Zekâ tabanlı bir sistem için test senaryoları elde etmek ve yürütmek için ikili testler uygular.

9.3 Sırt Sırta Test Etme

YZ-9.3.1 (K2) Yapay Zekâ tabanlı sistemler için sırt sırta testin nasıl kullanıldığını açıklar

9.4 A/B Testi

YZ-9.4.1 (K2) Yapay Zekâ tabanlı sistemlerin test edilmesinde A/B testinin nasıl uygulandığını açıklar.

9.5 Metamorfik Testler

YZ-9.5.1 (K3) Yapay Zekâ tabanlı sistemlerin testi için metamorfik test uygular.

HO-9.5.1 (H2) Belirli bir senaryo için test senaryoları elde etmek ve bunları yürütmek için metamorfik test uygular.

9.6 Yapay Zekâ Tabanlı Sistemlerin Tecrübeye Dayalı Testi

YZ-9.6.1 (K2) Tecrübeye dayalı testin Yapay Zekâ tabanlı sistemlerin testine nasıl uygulanabileceğini açıklar.

HO-9.6.1 (H2) Yapay Zekâ tabanlı bir sisteme keşif testi uygular.

9.7 Yapay Zekâ Tabanlı Sistemler için Test Tekniklerinin Seçilmesi

YZ-9.7.1 (K4) Belirli bir senaryo için, Yapay Zekâ tabanlı bir sistemi test ederken uygun test tekniklerini seçer.

9.1 Karşıt Saldırıları ve Veri Zehirlenmesi

9.1.1 Karşıt Saldırıları

Karşıt saldırısı, bir saldırganın eğitilmiş modele aktarılan geçerli girdileri kurnazca bozarak modelin yanlış tahminler sağlamasına neden olduğu saldıdır. Karşıt örnekler olarak bilinen bu bozulmuş girdiler ilk olarak, okunabilirliği kaybetmeden bir spam e-postayı hafifçe değiştirerek kandırılabilen spam filtrelerinde fark edilmiştir. Son zamanlarda, görüntü sınıflandırıcılarla daha ilişkili hale gelmişlerdir. İnsan gözüyle görülemeyen birkaç pikseli değiştirerek, bir sinir ağını görüntü sınıflandırmasını çok farklı bir nesneyle ve yüksek derecede güvenle değiştirmeye ikna etmek mümkündür.

Karşıt örnekler genellikle transfer edilebilir [B17], bu da bir MÖ sisteminin başarısız olmasına neden olan bir karşıt örneğin, aynı görevi gerçekleştirmek üzere eğitilmiş başka bir MÖ sisteminin de genellikle başarısız olmasına neden olacağı anlamına gelir. İkinci Makine Öğrenimi sistemi farklı verilerle eğitilmiş ve farklı mimarilere dayanıyor olsa bile, genellikle aynı karşıt örneklerle başarısız olmaya eğilimlidir.

Beyaz kutu karşıt saldırıları, saldırganın modeli eğitmek için hangi algoritmanın kullanıldığını ve ayrıca hangi model ayarlarının ve parametrelerinin kullanıldığını bildiği saldırılardır (makul düzeyde şeffaflık vardır). Saldırgan bu bilgiyi, örneğin girdilerde küçük değişiklikler yaparak ve hangilerinin model çıktılarında büyük değişikliklere neden olduğunu izleyerek karşıt örnekler oluşturmak için kullanır.

Kara kutu karşıt saldırıları, saldırganın işlevselliğini belirlemek için modeli keşfetmesini ve ardından benzer işlevsellik sağlayan kopya bir model oluşturmasını içerir. Saldırgan daha sonra bu kopya modele yönelik karşıt örnekleri belirlemek için beyaz kutu yaklaşımını kullanır. Karşıt örnekler genellikle aktarılabilir olduğundan, aynı karşıt örnekler normalde orijinal model üzerinde de çalışacaktır.

Kopya bir model oluşturmak mümkün değilse, farklı karşıt örnekleri keşfetmek ve sonuçları gözlemlemek için yüksek hacimli otomatik testler kullanmak mümkün olabilir.

Karşıt testler basitçe, güvenlik açıklarını tespit etmek amacıyla karşıt saldırılar gerçekleştirerek gelecekteki hatalara karşı koruma sağlamak için önleyici tedbirler alınmasını amaçlar. Belirlenen karşıt örnekler, modelin onları doğru bir şekilde tanıması için eğitim verilerine eklenir.

9.1.2 Veri Zehirlenmesi

Veri zehirlenme saldırıları, bir saldırganın iki sonuçtan birini elde etmek için eğitim verilerini manipüle ettiği saldırılardır. Saldırgan, gelecekteki izinsiz girişleri kolaylaştırmak için arka kapılar veya sinir ağı Truva atları ekleyebilir veya daha sık olarak, eğitilmiş modelin yanlış tahminler sağlamasına neden olmak için bozuk eğitim verilerini (örneğin, yanlış etiketlenmiş veriler) kullanırlar.

Zehirlenme saldırıları, Makine Öğrenimi sisteminin belirli durumlarda yanlış sınıflandırmasına neden olmak amacıyla hedeflenebilir. Servis dışı bırakma saldırısında olduğu gibi gelişigüzel de olabilirler. Zehirlenme saldırısının iyi bilinen bir örneği, Microsoft Tay sohbet robotunun bozulmasıydı; bu sayede nispeten az sayıda zararlı Twitter konuşması, gelecekte kusurlu konuşmalar sağlamak için geri bildirim yoluyla sistemi eğitti. Veri zehirlenme saldırılarının sıklıkla kullanılan bir biçimi, spam filtreleme yazılımını çarpıtmak amacıyla milyonlarca spam e-postasının spam değilmiş gibi yanlış raporlanmasıdır. Veri zehirlenmesiyle ilgili bir endişe alanı da herkese açık, yaygın olarak kullanılan Yapay Zekâ veri setlerinin zehirlenme potansiyelidir.

Zehirlenmiş veriler aykırı değerler olarak görünebileceğinden, veri zehirlenme saldırısını tespit etmek için EDA (Keşifsel Veri Analizi) kullanılarak test yapmak mümkündür. Ayrıca, eğitim verilerinin kaynağından emin olmak için veri toplama politikaları gözden geçirilebilir. Operasyonel bir Makine Öğrenimi sisteminin zehirli verilerle beslenerek saldırıya uğrayabileceği durumlarda, sistemin güncellenmiş sürümünün önceki sürümle hala yakından uyumlu olup olmadığını kontrol etmek için A/B testi (bkz. Bölüm 9.4) kullanılabilir. Alternatif olarak,

güvenilir bir test grubu kullanılarak güncellenmiş bir sistemin regresyon testi de bir sistemin zehirlenip zehirlenmediğini belirleyebilir.

9.2 İkili Testler

Yapay Zekâ tabanlı bir sistem için ilgilenen parametrelerin sayısı, özellikle sistem büyük veri kullandığında veya otonom bir araç gibi dış dünyayla etkileşime girdiğinde son derece yüksek olabilir. Geniş kapsamlı test, bu parametrelerin tüm olası değerlere ayarlanmış tüm olası kombinasyonlarının test edilmesini gerektirir. Ancak, bu pratikte sonsuz sayıda testle sonuçlanacağından, mevcut sınırlı sürede çalıştırılabilecek bir alt küme seçmek için test teknikleri kullanılır.

Her biri çok sayıda ayırık değere sahip olabilen çok sayıda parametrenin birleştirilmesinin mümkün olduğu durumlarda, ideal olarak test grubunun hata tespit yeteneğinden ödün vermeden gerekli test senaryolarının sayısını önemli ölçüde azaltmak için kombinasyonlu test etme uygulanabilir. Çeşitli kombinasyonlu test etme teknikleri vardır (bkz. [I02] ve [S08]). Bununla birlikte, pratikte, ikili test en yaygın kullanılan tekniktir çünkü anlaşılması kolaydır ve geniş araç desteğine sahiptir. Buna ek olarak, araştırmalar çoğu hatanın az sayıda parametre içeren etkileşimlerden kaynaklandığını göstermiştir [B33].

Pratikte, ikili testlerin kullanılması bile bazı sistemler için kapsamlı test gruplarına neden olabilir ve gerekli sayıda testin çalıştırılmasına izin vermek için genellikle otomasyon ve sanal test ortamlarının (bkz. Bölüm 10.2) kullanılması gerekli hale gelir. Örneğin, otonom araçlar düşünüldüğünde, sistem testi için üst düzey test senaryolarının hem otomobillerin çalışması beklenen farklı ortamları hem de çeşitli araç işlevlerini çalıştırması gerekir. Bu nedenle, parametrelerin çeşitli ortam kısıtlamalarını (örn. yol türleri ve yüzeyleri, hava ve trafik koşulları ve görüş mesafesi) ve çeşitli otonom sürüş işlevlerini (örn. uyarlanabilir hız sabitleyici, şeritte tutma yardımı ve şerit değiştirme yardımı) içermesi gerekecektir. Bu parametrelere ek olarak, sensörlerden gelen girdiler değişen etkinlik seviyelerinde dikkate alınabilir (örneğin, bir video kameradan gelen girdiler yolculuk ilerledikçe ve yol kirlendikçe azalacaktır).

Otonom araçlar gibi güvenlik açısından kritik Yapay Zekâ tabanlı sistemlerde kombinasyonlu test etme kullanımı için gereken titizlik seviyesi araştırma açısından henüz net değildir. İkili testler yeterli olmasa da bu yaklaşımın hataları bulmada etkili olduğu bilinmektedir.

9.2.1 Uygulamalı Egzersiz: İkili Testler

Minimum beş parametreye ve en az beş yüz olası kombinasyona sahip Yapay Zekâ tabanlı bir sistem için, azaltılmış bir ikili kombinasyon kümesi belirlemek ve bu kombinasyonlar için testleri yürütmek üzere bir ikili test aracı kullanın. Test edilen ikili kombinasyon sayısını, teorik olarak mümkün olan tüm kombinasyonların test edilmesi durumunda gereken sayı ile karşılaştırın.

9.3 Sırt Sırta Test Etme

Yapay Zekâ tabanlı sistemleri test ederken test sonucunu bilen sorununa (bkz. Bölüm 8.7) yönelik potansiyel çözümlerden biri, sırt sırta test kullanmaktır. Bu aynı zamanda diferansiyel test olarak da bilinir. Sırt sırta testlerle, sistemin alternatif bir versiyonu sahte bir doğrulayıcı olarak kullanılır ve çıktıları, test edilen sistemin (SUT) ürettiği test sonuçlarıyla karşılaştırılır. Sözde sonucu bilen, mevcut bir sistem olabilir veya farklı bir ekip tarafından, muhtemelen farklı bir platformda, farklı bir mimariyle ve farklı bir programlama diliyle geliştirilebilir. Fonksiyonel kullanılabilirlik test edilirken (Fonksiyonel olmayan gereksinimlerin aksine), sözde sonucu bilen olarak kullanılan sistem, SUT ile aynı fonksiyonel olmayan kabul kriterlerine ulaşmakla kısıtlı değildir. Örneğin, o kadar hızlı yürütülmesi gerekmeyebilir, bu durumda inşa edilmesi çok daha ucuz olabilir.

Makine Öğrenimi bağlamında, bir Makine Öğrenimi sözde sonucu bilen oluşturmak için farklı çerçeveler, algoritmalar ve model ayarları kullanmak mümkündür. Bazı durumlarda, geleneksel, Yapay Zekâ olmayan yazılım kullanarak bir sözde sonucu bilen oluşturmak da mümkün olabilir.

Sözde sonucu bilen hataları tespit etmede etkili olabilmesi için hem sözde sonucu bilen hem de test edilen sistemde (SUT) ortak bir yazılım bulunmamalıdır. Aksi takdirde, her ikisi de kusurlu olduğunda aynı hatanın iki test sonucunun eşleşmesine neden olması mümkün olacaktır. Yapay Zekâ tabanlı sistemler geliştirmek için çok fazla olgunlaşmamış, yeniden kullanılabilir, açık kaynaklı Yapay Zekâ yazılımı kullanıldığından, kodun sözde sonucu bilen ile SUT (test edilen sistem) arasında yeniden kullanılması, sözde sonucu bilen tehlikeye atabilir. Yeniden kullanılabilir Yapay Zekâ çözümlerinin yetersiz dokümantasyonu, test uzmanlarının bu sorunun ortaya çıktığını fark etmesini de zorlaştırabilir.

9.4 A/B Testi

A/B testi, programın iki değişkeninin (A ve B) aynı girdilere verdiği yanıtın, iki değişkenden hangisinin daha iyi olduğunu belirlemek amacıyla karşılaştırıldığı bir yöntemdir. Programlar arasındaki farkı belirlemek için genellikle çeşitli test çalıştırmalarından elde edilen test sonuçlarının karşılaştırılmasını gerektiren istatistiksel bir test yaklaşımıdır.

Bu yöntemin basit bir örneği, iki gruba ayrılmış bir pazarlama listesine iki promosyon teklifinin e-posta ile gönderilmesidir. Listenin yarısı A teklifini, yarısı da B teklifini alır ve her bir teklifin başarısı gelecekte hangisinin kullanılacağına karar verilmesine yardımcı olur. Birçok e-ticaret ve web tabanlı şirket, tüketicilerin tercihlerini belirlemeye yardımcı olmak için farklı tüketicileri farklı işlevlere yönlendirerek üretimde A/B testi kullanır.

A/B testi, mevcut sistemin kısmi bir sonucu bilen olarak kullanıldığı test sonucunu bilen problemini çözmeye yönelik bir yaklaşımdır. A/B testi, test senaryoları oluşturmaz ve testlerin nasıl tasarlanması gerektiği konusunda rehberlik sağlamaz; ancak testlerde genellikle operasyonel girdiler kullanılır.

A/B testi, Bölüm 5'te açıklandığı gibi Makine Öğrenimi fonksiyonel performans metrikleri gibi üzerinde anlaşmaya varılmış kabul kriterlerinin bulunduğu Yapay Zekâ tabanlı bir sistemdeki güncellemeleri test etmek için kullanılabilir. Sistem her güncellendiğinde, güncellenen varyantın önceki varyant kadar iyi veya ondan daha iyi performans gösterip göstermediğini kontrol etmek için A/B testi kullanılır. Böyle bir yaklaşım basit bir sınıflandırıcı için kullanılabileceği gibi çok daha karmaşık sistemleri test etmek için de kullanılabilir. Örneğin, bir akıllı şehir ulaşım yönlendirme sisteminin etkinliğini artırmaya yönelik bir güncelleme A/B testi kullanılarak da test edilebilir (örneğin, birbirini takip eden haftalarda sistemin iki varyantı için ortalama işe gidip gelme sürelerinin karşılaştırılması).

A/B testi kendi kendine öğrenen sistemleri test etmek için de kullanılabilir. Sistem bir değişiklik yaptığında otomatik testler yapılır ve ortaya çıkan sistem özellikleri, değişiklik yapılmadan önceki özelliklerle karşılaştırılır. Sistem iyileştirilirse, değişiklik kabul edilir, aksi takdirde sistem önceki durumuna geri döner.

9.5 Metamorfik Test (MT)

A/B testi ile sırt sırta test arasındaki önemli bir fark, A/B testinin aynı sistemin iki varyantını karşılaştırmak için kullanılması ve sırt sırta testin hataları tespit etmek için kullanılmasıyla ilgilidir.

Metamorfik test [B18], geçmiş bir kaynak test senaryosunu temel alan test senaryoları oluşturmayı amaçlayan bir tekniktir. Bir veya daha fazla takip test senaryosu, bir metamorfik ilişkiye (MR) dayalı olarak kaynak test senaryosunu değiştirerek (başkalaştırarak) oluşturulur. Metamorfik ilişki, test nesnesinin gerekli bir fonksiyonun bir özelliğine dayanır, öyle ki bir test senaryosunun test girdilerindeki bir değişikliğin aynı test senaryosunun beklenen sonuçlarına nasıl yansıdığını açıklar.

Örneğin, bir dizi sayının ortalamasını belirleyen bir program düşünün. Bir dizi sayı ve beklenen ortalamayı içeren bir kaynak test senaryosu oluşturulur ve test senaryosu başarılı olduğunu

doğrulamak için çalıştırılır. Artık programın ortalama fonksiyonu hakkında bilinenlere dayalı olarak takip test senaryoları oluşturmak mümkündür. Başlangıçta, ortalaması alınan sayıların sırası basitçe değiştirilebilir. Ortalama fonksiyonu göz önüne alındığında, beklenen sonucun aynı kalacağı tahmin edilebilir.

Böylece, beklenen sonucun hesaplanmasına gerek kalmadan sayıların farklı sıralarda olduğu bir takip test senaryosu oluşturulabilir. Büyük bir sayı kümesi söz konusu olduğunda bu, aynı sayıların farklı sıralarda kullanıldığı çok sayıda farklı sayı kümesinin oluşturulmasına yol açabilir ve her biri ayrı bir takip test senaryosu oluşturmak için kullanılabilir. Tüm bu test senaryoları aynı kaynak test senaryosunu temel alır ve aynı beklenen sonuca sahip olur.

Beklenen sonucun kaynak test senaryosunun orijinal beklenen sonucundan farklı olduğu metamorfik ilişkilere ve takip test senaryolarına sahip olmak yaygındır. Örneğin, aynı ortalama fonksiyonu kullanılarak, giriş kümesinin her bir elemanının iki ile çarpıldığı bir metamorfik ilişki türetilebilir. Böyle bir küme için beklenen sonuç, basitçe orijinal beklenen sonucun iki ile çarpımıdır. Benzer şekilde, bu metamorfik ilişkiye dayalı potansiyel olarak sonsuz sayıda takip test senaryosu oluşturmak için çarpan olarak başka herhangi bir değer kullanılabilir.

Metamorfik test çoğu test nesnesi için kullanılabilir ve hem fonksiyonel hem de fonksiyonel olmayan testlere uygulanabilir (örneğin, kurulum testi, kurulum parametrelerinin farklı sıralarda seçilebildiği farklı hedef konfigürasyonları kapsar). Ucuz bir test sonucunu bilenin olmaması nedeniyle beklenen sonuçların üretilmesinin sorunlu olduğu durumlarda özellikle kullanışlıdır. Bu durum, büyük verilerin analizine dayanan bazı Yapay Zekâ tabanlı sistemlerde veya test uzmanlarının Makine Öğrenimi algoritmasının tahminlerini nasıl elde ettiği konusunda net olmadığı durumlarda söz konusudur. Yapay Zekâ alanında metamorfik test, diğerlerinin yanı sıra görüntü tanıma, arama motorları, rota optimizasyonu ve ses tanımayı test etmek için kullanılmıştır.

Yukarıda açıklandığı gibi, metamorfik test başarılı bir kaynak test senaryosuna dayanabilir, ancak herhangi bir kaynak test senaryosunun doğru olduğunu doğrulamanın mümkün olmadığı durumlarda da yararlıdır. Bu durum, örneğin programın, bazı Yapay Zekâ tabanlı sistemlerde olduğu gibi, bir insan test uzmanının kopyalayamayacağı ve test sonucunu bilen olarak kullanamayacağı kadar karmaşık bir fonksiyon uyguladığı durumlarda söz konusu olabilir. Bu durumda metamorfik test, çalıştırıldığında çıktılar arasındaki ilişkilerin geçerliliğinin kontrol edilebileceği bir dizi çıktı oluşturacak bir veya daha fazla test senaryosu oluşturmak için kullanılabilir. Bu tür bir metamorfik test ile, bireysel testlerin doğru olduğu bilinmez, ancak aralarındaki ilişkilerin doğru olması gerekir, bu sayede programa olan güven artar. Buna bir örnek olarak, geniş bir veri setine dayanarak ölüm yaşını tahmin eden Yapay Zekâ tabanlı bir aktüerya programı verilebilir; burada, örneğin, içilen sigara sayısı artırılırsa, tahmin edilen ölüm yaşının düşmesi (veya en azından aynı kalması) gerektiği bilinmektedir.

Metamorfik test nispeten yeni bir test tekniğidir, ilk olarak 1998'de önerilmiştir. Geleneksel test tekniklerinden farkı, takip test senaryolarının beklenen sonuçlarının mutlak değerler cinsinden değil, kaynak test senaryosunda beklenen sonuçlara göre göreceli olarak tanımlanmasıdır. Kolay anlaşılır bir konsepte dayanır, tekniği uygulama konusunda çok az deneyimi olan ancak uygulama alanını anlayan test uzmanları tarafından uygulanabilir ve geleneksel tekniklere kıyasla benzer maliyetlere sahiptir. Ayrıca hataların ortaya çıkarılmasında da etkilidir; araştırmalar, yalnızca üç ila altı farklı metamorfik ilişkinin geleneksel bir test sonucunu bilene dayalı teknikler kullanılarak tespit edilebilecek hataların %90'ından fazlasını ortaya çıkarabildiğini göstermektedir [B19]. İyi tanımlanmış metamorfik ilişkilere ve bir kaynak test senaryosundan otomatik olarak takip test senaryoları oluşturmak mümkündür. Bununla birlikte, ticari araçlar şu anda mevcut değildir, ancak Google, açık kaynaklı Graphics Fuzz aracını kullanarak Android grafik sürücülerini test etmek için otomatik metamorfik testi zaten uygulamaktadır (bkz. [R23]).

9.5.1 Uygulamalı Egzersiz: Metamorfik Test

Bu egzersizde öğrenciler aşağıdaki konularda pratik deneyim kazanacaklardır:

- Belirli bir Yapay Zekâ tabanlı uygulama veya program için çeşitli metamorfik ilişkilerin (MR'ler) türetilmesi. Bu metamorfik ilişkiler, kaynak ve takip testi senaryolarının beklenen sonuçlarının bazılarını aynı, bazılarını ise farklı olan durumları içermelidir.
- Yapay Zekâ tabanlı uygulama veya program için kaynak test senaryoları oluşturma. Bunların geçmesinin garanti edilmesi gerekmez, ancak böyle bir "altın standardın" mevcut olmadığı durumlarda öğrencilere metamorfik testin sınırlamaları hatırlatılmalıdır.
- Takip test senaryoları üretmek için türetilen metamorfik ilişkileri ve oluşturulan kaynak test senaryolarını kullanma.
- Takip test senaryolarının çalıştırılması.

9.6 YZ Tabanlı Sistemlerin Tecrübeye Dayalı Testi

Tecrübeye dayalı testler hata tahminleme, keşif testi ve kontrol listesine dayalı testi içerir [I01], bunların tümü Yapay Zekâ tabanlı sistemlerin testine uygulanabilir.

Hata tahminleme genellikle test uzmanlarının bilgisine, tipik geliştirici hatalarına ve benzer sistemlerdeki (veya önceki sürümlerdeki) hatalara dayanır. Yapay Zekâ tabanlı sistemlere uygulanan hata tahminlemeye bir örnek, sistematik olarak yanlışlık içeren eğitim verilerinin kullanılması nedeniyle Makine Öğrenimi sistemlerinin geçmişte nasıl başarısız olduğu hakkındaki bilginin kullanılması olabilir.

Keşif testinde, testler yinelemeli bir şekilde tasarlanır, oluşturulur ve yürütülür; önceki testlerin test sonuçlarına dayalı olarak sonraki testlerin türetilmesi için fırsat sağlanır. Keşif testi, özellikle Yapay Zekâ tabanlı sistemlerde sıklıkla görülen zayıf gereksinimler veya test sonucunu bilen problemleri olduğunda faydalıdır. Sonuç olarak, keşif testi bu bağlamda sıklıkla kullanılır ve metamorfik test gibi tekniklere dayalı daha sistematik testleri desteklemek için kullanılmalıdır (bkz. Bölüm 9.5).

Tur, test uzmanlarının özel bir odak etrafında düzenlenmiş keşif testi gerçekleştirirken başvurabilecekleri bir dizi strateji ve hedef için kullanılan bir metaforudur [B20]. Yapay Zekâ tabanlı sistemlerin keşif testi için tipik turlar, Makine Öğrenimi sistemlerinde yanlışlık, eksik öğrenme ve aşırı öğrenme kavramlarına odaklanabilir. Örneğin, modeli test etmek için bir veri turu uygulanabilir. Bu turda test uzmanı, eğitim için kullanılan farklı veri türlerini, bunların dağılımını, varyasyonlarını, formatlarını ve aralıklarını vb. belirleyebilir ve ardından modeli test etmek için veri türlerini kullanabilir.

MÖ sistemleri büyük ölçüde eğitim verilerinin kalitesine bağlıdır ve mevcut EDA (Keşifsel Veri Analizi) alanı keşif testi yaklaşımıyla yakından ilgilidir. EDA (Keşifsel Veri Analizi), verilerin örüntüler, ilişkiler, eğilimler ve aykırı değerler açısından incelendiği yerdir. Verilerin etkileşimli, hipotez odaklı keşfini içerir ve [B21]'de "Verileri beklentilerle keşfederiz. Verilerde gördüklerimize dayanarak beklentilerimizi gözden geçiririz. Ve bu süreci yineleriz." EDA (Keşifsel Veri Analizi) tipik olarak iki alanda araç desteğine ihtiyaç duyar; analistlerin karmaşık verileri daha iyi anlamalarını sağlamak için verilerle etkileşim ve analiz sonuçlarını kolayca görüntülemelerini sağlamak için veri görselleştirme. Veri görselleştirme tarafından başlıca yönlendirilen keşif tekniklerin kullanımı, kullanılan Makine Öğrenimi algoritmasını doğrulamaya, verimli modellere yol açan değişiklikleri belirlemeye ve alan uzmanlığını kullanmaya yardımcı olabilir [B22].

Google, Makine Öğrenimi sistemleri için Google bünyesinde bir test kontrol listesi olarak kullanılan veri, model geliştirme, altyapı ve izleme alanlarında iddialar olarak yazılmış yirmi sekiz Makine Öğrenimi testinden oluşan bir sete sahiptir [B23]. Google "Makine Öğrenimi test kontrol listesi" burada Google tarafından yayınlandığı şekliyle sunulmaktadır:

Makine Öğrenimi Verileri:

1. Özellik beklentileri bir şemada yakalanır.
2. Tüm özellikler faydalıdır.
3. Hiçbir özelliğin maliyeti çok fazla değildir.
4. Özellikler üst düzey gerekliliklere uygundur.
5. Veri ardışık düzeni uygun gizlilik kontrollerine sahiptir.
6. Yeni özellikler hızlı bir şekilde eklenebilir.
7. Tüm giriş özellik kodu test edilmiştir.

Model Geliştirme:

1. Model özellikleri gözden geçirilir ve gönderilir.
2. Çevrimdışı ve çevrimiçi metrikler birbiriyle ilişkilidir.
3. Tüm hiper-parametreler ayarlanmıştır.
4. Model eskiliğinin etkisi bilinmektedir.
5. Daha basit bir model daha iyi değildir.
6. Önemli veri dilimlerinde model kalitesi yeterlidir.
7. Model, kapsayıcılık hususları açısından test edilmiştir.

Makine Öğrenimi Altyapısı:

1. Eğitim tekrarlanabilir.
2. Model özellikleri birim olarak test edilmiştir.
3. Makine Öğrenimi ardışık düzeni entegrasyon testinden geçirilmiştir.
4. Model kalitesi servis edilmeden önce doğrulanır.
5. Modelde hata ayıklanabilir.
6. Modeller servis edilmeden önce kanarya testi yapılır.
7. Servis modelleri geri alınabilir.

İzleme Testleri:

1. Bağımlılık değişiklikleri bildirimle sonuçlanır.
2. Veri değişmezleri girdiler için geçerlidir.
3. Eğitim ve servis çarpık değildir.
4. Modeller çok eski değil.

5. Modeller sayısal olarak sabittir.
6. Bilgi işlem performansı gerilememiştir.
7. Tahmin kalitesi gerilememiştir.

9.6.1 Uygulamalı Egzersiz: Keşif Testi ve Keşifsel Veri Analizi (EDA)

Seçilen bir model ve veri seti için öğrenciler, çeşitli veri türlerini ve bunların çeşitli parametreler için dağılımlarını göz önünde bulundurarak bir veri turu gerçekleştireceklerdir.

Öğrenciler, eksik verileri ve/veya verilerdeki potansiyel yanlışlıkları belirlemek için veriler üzerinde EDA (Keşifsel Veri Analizi) gerçekleştireceklerdir.

9.7 YZ Tabanlı Sistemler için Test Tekniklerinin Seçilmesi

Yapay Zekâ tabanlı bir sistem tipik olarak hem Yapay Zekâ hem de Yapay Zekâ olmayan bileşenleri içerecektir. Yapay Zekâ olmayan bileşenleri test etmek için test tekniklerinin seçimi genellikle herhangi bir geleneksel testle aynıdır. Yapay Zekâ tabanlı bileşenler için seçim daha kısıtlı olabilir. Örneğin, bir test sonucunu bilen probleminin algılandığı durumlarda (yani, beklenen sonuçların üretilmesinin zor olduğu durumlarda), algılanan risklere bağlı olarak, bu problemi aşağıdakileri kullanarak azaltmak mümkündür:

- Sırt sırta test: Bu, test senaryolarının mevcut olmasını veya oluşturulmasını ve regresyon testi için sistemin önceki bir sürümü olabilecek sözde bir sonucu bilen görevi göreceğ eşdeğer bir sistem gerektirir. Hataların etkili bir şekilde tespit edilmesi için bağımsız olarak geliştirilmiş bir sistem gerekebilir.
- A/B testi: Bu test genellikle operasyonel girdileri test senaryoları olarak kullanır ve normalde istatistiksel analiz kullanarak aynı sistemin iki varyantını karşılaştırmak için kullanılır. A/B testi, yeni bir varyantın veri zehirlenmesini kontrol etmek veya kendi kendine öğrenen bir sistemin otomatik regresyon testi için kullanılabilir.
- Metamorfik test: Bu, deneyimsiz test uzmanları tarafından, uygulama alanını anlamaları gerekmesine rağmen, hataları uygun maliyetli bir şekilde bulmak için kullanılabilir. Metamorfik test, beklenen sonuçlar mutlak olmadığından, bunun yerine kaynak test senaryolarına göre göreceli olduğundan kesin sonuçlar sağlamak için uygun değildir. Ticari araç desteği şu anda mevcut değildir, ancak birçok test manuel olarak oluşturulabilir.

Karşıt testler tipik olarak, karşıt örneklerin yanlış kullanımının önemli bir etkiye sahip olabileceği veya sistemin saldırıya uğrayabileceği Makine Öğrenimi modelleri için uygundur. Benzer şekilde, veri zehirlenmesi testi, sistemin saldırıya uğrayabileceği Makine Öğrenimi sistemleri için uygun olabilir.

Yapay Zekâ tabanlı sistemlerin karmaşık olduğu ve birden fazla parametreye sahip olduğu durumlarda, ikili testler genellikle uygundur.

Tecrübeye dayalı testler, özellikle eğitim ve operasyonel veriler için kullanılan verilerin dikkate alınması açısından Yapay Zekâ tabanlı sistemlerin test edilmesi için genellikle uygundur. EDA (Keşifsel Veri Analizi), kullanılan Makine Öğrenimi algoritmasını doğrulamak, verimlilik iyileştirmelerini belirlemek ve alan uzmanlığından yararlanmak için kullanılabilir. Google, Makine Öğrenimi test kontrol listesinin, Makine Öğrenimi sistemleri için etkili bir yaklaşım olduğunu tespit etmiştir.

Sinir ağlarının özel alanında, ağın kapsamı genellikle görev açısından kritik sistemler için uygundur ve bazı kapsam kriterleri diğerlerine göre daha titiz bir kapsam gerektirir.

10.YZ Tabanlı Sistemler için Test Ortamları - 30 Dakika

Anahtar Kelimeler

Sanal Test Ortamı

YZ-Spesifik Anahtar Kelimeleri

Yapay Zekâ'ya Özgü İşlemci, Otonom Sistem, Büyük Veri, Açıklanabilirlik, Çoklu Ajan Sistemi, Kendi Kendine Öğrenen Sistem

Bölüm 10 için Öğrenme Hedefleri:

10.1 Yapay Zekâ Tabanlı Sistemler için Test Ortamları

YZ-10.1.1 K2 Yapay Zekâ tabanlı sistemler için gereken test ortamlarını geleneksel sistemlerden ayıran ana faktörleri açıklar.

10.2 Yapay Zekâ Tabanlı Sistemleri Test Etmek için Sanal Test Ortamları

YZ-10.2.1 K2 Sanal test ortamlarının Yapay Zekâ tabanlı sistemlerin testinde sağladığı faydaları açıklar.

10.1 YZ Tabanlı Sistemler için Test Ortamları

Yapay Zekâ tabanlı sistemler geniş bir operasyonel çevrede kullanılabilir, bu da test ortamlarının benzer şekilde çeşitli olmasını sağlar. Geleneksel sistemler için gereken test ortamlarından farklılık gösterebilecek Yapay Zekâ tabanlı sistemlerin özellikleri şunları içerir:

- Kendi kendine öğrenme: Kendi kendine öğrenen sistemler ve bazı otonom sistemlerin, sistem başlangıçta dağıtıldığında tam olarak tanımlanmamış olabilecek değişen operasyonel ortamlara uyum sağlaması beklenir (Bkz. Bölüm 2.1). Sonuç olarak, bu tanımlanmamış çevresel değişiklikleri taklit edebilen test ortamlarını tanımlamak, test uzmanlarının hayal gücünü ve test ortamına rastlantısal bir seviye dahil etmeyi gerektirebilir.
- Otonomi: Otonom sistemlerin, insan müdahalesi olmadan çevrelerindeki değişikliklere yanıt vermesi ve ayrıca otonominin insan operatörlere geri verilmesi gereken durumları tanıması beklenir (Bkz. Bölüm 2.2). Bazı sistemler için otonomi verme koşullarını belirlemek ve taklit etmek, sistemleri uç noktalara itmek için test ortamlarının oluşturulması gerekebilir. Bazı otonom sistemler için, amaçları tehlikeli ortamlarda çalışmaktır ve temsilci, tehlikeli test ortamlarının oluşturulması zor olabilir.
- Çoklu ajan: Çoklu ajanlı Yapay Zekâ tabanlı sistemlerin, diğer Yapay Zekâ tabanlı sistemlerle birlikte çalışması bekleniyorsa; test edilen sistemin (SUT) etkileşimde bulunduğu Yapay Zekâ tabanlı sistemlerin belirsizliğini taklit edebilmesi için test ortamının bir düzeyde belirsizlik içermesi gerekebilir.
- Açıklanabilirlik: Bazı Yapay Zekâ tabanlı sistemlerin doğru kararlarını nasıl verdiğini belirlemek zor olabilir (Bkz. Bölüm 2.7). Bu, dağıtımdan önce anlaşılması önemli olduğunda, kararların nasıl alındığını açıklamak için araçları içeren bir test ortamı gerekebilir.
- Donanım: Yapay Zekâ tabanlı sistemleri barındırmak için kullanılan bazı donanımlar özellikle bu amaç için tasarlanmıştır, örneğin Yapay Zekâ özel işlemcileri (Bkz. Bölüm 1.6). İlgili test planlamanın bir parçası olarak bu tür donanımın test ortamına dahil edilmesi gerekliliği göz önünde bulundurulmalıdır.
- Büyük veri: Bir Yapay Zekâ tabanlı sistemin büyük veri tüketmesi bekleniyorsa (örneğin, yüksek hacimli, yüksek hızda ve/veya yüksek çeşitlilikte veri), bu verinin bir test ortamının bir parçası olarak kurulması dikkatli planlama ve uygulama gerektirir (Bkz. Bölüm 7.3).

10.2 YZ Tabanlı Sistemleri Test Etmek için Sanal Test Ortamları

Bir Yapay Zekâ tabanlı sistemi test etmek için sanal bir test ortamı kullanmanın getirdiği faydalar şunlardır:

- Tehlikeli senaryolar: Bu senaryolar, SUT'i, diğer etkileşimde bulunan sistemleri (insanlar dahil) veya operasyonel ortamı (örneğin, ağaçlar, binalar) tehlikeye atmadan test edilebilir.
- Sıra dışı senaryolar: Bu senaryolar, aksi takdirde gerçek işlemler için bu senaryoların oluşturulması çok zaman alıcı veya maliyetli olduğunda test edilebilir (örneğin, nadir bir olayı beklemek, güneş tutulması veya dört otobüsün aynı kavşağa aynı anda girmesi gibi). Benzer şekilde, gerçek dünyada zor oluşturulan sınır durumları, sanal bir test ortamında daha kolay, tekrarlanabilir ve tekrarlanabilir şekilde oluşturulabilir.
- Aşırı senaryolar: Bu senaryolar, bunları gerçek hayatta kurmanın pahalı veya imkansız olduğu durumlarda test edilebilir (örneğin, nükleer bir felaket veya derin uzay keşfi için)...

- Zaman alıcı senaryolar: Bu senaryolar, sanal bir ortamda (örneğin, saniyede birkaç kez) azaltılmış zaman dilimlerinde test edilebilir. Buna karşılık, bu senaryoların gerçek zamanlı olarak kurulması ve çalıştırılması saatler veya günler alabilir. Başka bir avantaj ise birden fazla sanal test ortamının paralel olarak çalıştırılabilmesidir. Bu genellikle bulutta gerçekleşir ve birçok senaryonun eşzamanlı olarak çalıştırılmasına olanak tanır, bu, gerçek sistem donanımını kullanarak mümkün olmayabilir.
- Gözlemlenebilirlik ve kontrol edilebilirlik: Sanal test ortamları, test ortamının çok daha fazla kontrol edilebilirliğini sağlar. Örneğin, olağandışı bir finansal işlem koşul setinin çoğaltılmasını sağlayabilirler. Ayrıca, tüm dijital olarak sağlanan ortam bileşenleri sürekli olarak izlenebilir ve kaydedilebilir.
- Kullanılabilirlik: Sanal test ortamları tarafından donanımın simülasyonu, belki henüz geliştirilmemiş veya maliyetli oldukları için mevcut olmayan (simüle edilmiş) donanım bileşenleri ile sistemlerin test edilmesine olanak tanır.

Sanal test ortamları, belirli bir sisteme özel olarak inşa edilebilir, genel veya belirli uygulama alanlarını desteklemek için geliştirilebilir. Ticari ve açık kaynaklı sanal test ortamları, Yapay Zekâ tabanlı sistemlerin testini desteklemek için mevcuttur. Örnekler şunları içerir:

- Morse: Modüler Açık Robot Simülasyon Motoru, Blender oyun motorunu [R24] temel alan, tekli veya çoklu robotların genel mobil robot simülasyonuna yönelik bir simülatördür.
- YZ Habitat: Bu, Facebook YZ tarafından oluşturulan ve somutlaştırılmış araçları (sanal robotlar gibi) foto-gerçekçi 3D ortamlarda eğitmek için tasarlanmış bir simülasyon platformudur [R25].
- DRIVE Constellation: Bu, NVIDIA'nın otonom araçlarına yönelik açık ve ölçeklenebilir bir platformdur. Bulut tabanlı bir platforma dayanmaktadır ve milyarlarca kilometrelik otonom araç testi oluşturma kapasitesine sahiptir [R26].
- MATLAB and Simulink: Bunlar, eğitim verileri hazırlama, Makine Öğrenimi modelleri üretme ve sentetik verileri kullanan modeller de dahil olmak üzere Yapay Zekâ tabanlı sistemlerin yürütülmesini simüle etme yeteneği sağlar [R27].

11. YZ ile Test Etme – 195 Dakika

Anahtar Kelimeler

Görsel Test Etme

YZ-Spesifik Anahtar Kelimeleri

Bayes Teknikleri, Sınıflandırma, Kümeleme Algoritması, Hata Tahmini, Grafik Kullanıcı Arayüzü (GUI)

Bölüm 11- Öğrenme Hedefleri:

11.1 Yazılım Testi için Yapay Zekâ Teknolojileri

YZ-11.1.1 K2 Yazılım testinde kullanılan Yapay Zekâ teknolojilerini sınıflandırır.

HO-11.1.1 H2 Yapay Zekânın daha az kullanılma olasılığı olan test etkinliklerini örnekler kullanarak tartışın.

11.2 Raporlanan Hataları Analiz Etmek için Yapay Zekâ Kullanımı

YZ-11.2.1 K2 Yapay Zekânın yeni hataların analizine nasıl destek verebileceğini açıklayın.

11.3 Test Senaryosu Oluşturma için Yapay Zekâ Kullanımı

YZ-11.3.1 K2 Yapay Zekânın test senaryosu oluşturmada nasıl yardımcı olabileceğini açıklayın.

11.4 Regresyon Test Gruplarının Optimizasyonu için Yapay Zekâ Kullanımı

YZ-11.4.1 K2 Yapay Zekânın regresyon test gruplarının optimizasyonunda nasıl yardımcı olabileceğini açıklayın.

11.5 Hata Tahmini için Yapay Zekâ Kullanımı

YZ-11.5.1 K2 Yapay Zekânın hata tahmininde nasıl yardımcı olabileceğini açıklayın.

HO-11.5.1 H2 Basit bir Yapay Zekâ tabanlı hata tahmin sistemi uygulayın.

11.6 Kullanıcı Arayüzlerini Test Etmek için Yapay Zekâ Kullanımı

YZ-11.6.1 K2 Yapay Zekânın kullanıcı arayüzlerini test etmede nasıl kullanıldığını açıklayın.

11.1 Yazılım Testi için YZ Teknolojileri

Birinci Bölüm'de, 1.4 kısmında listelenen birkaç Yapay Zekâ teknolojisi bulunmaktadır ve bunların her biri yazılım testinin belirli bir yönünü desteklemek için kullanılabilir. Harman'a göre [B24], yazılım mühendisliği topluluğu üç geniş Yapay Zekâ teknolojisi alanını kullanmaktadır:

- Bulanık mantık ve olasılıksal yöntemler: Bu yöntemler, kendileri olasılıksal olan gerçek dünya problemlerini ele almak için Yapay Zekâ tekniklerinin kullanılmasını içerir. Örneğin, Yapay Zekâ, Bayes tekniklerini kullanarak olası sistem arızalarını analiz etmek ve tahmin etmek için kullanılabilir. Bunlar, bileşenlerin veya fonksiyonların başarısız olma olasılığını tahmin edebilir veya sistemle insan etkileşimlerinin potansiyel olarak rastgele doğasını yansıtabilir.
- Sınıflandırma, öğrenme ve tahmin: Bu, proje planlaması kısmında maliyetleri tahmin etmek veya hataları tahmin etmek gibi çeşitli kullanım durumları için kullanılabilir. Makine Öğrenimi tarafından temsil edildiği gibi, bu alan birçok yazılım test görevi için kullanılmaktadır, bunlar arasında hata yönetimi (Bkz. Bölüm 11.2), hata tahmini (Bkz. Bölüm 11.5) ve kullanıcı arayüzü testi (Bkz. Bölüm 11.6) bulunmaktadır.
- Hesaplamalı arama ve optimizasyon teknikleri: Bu teknikler, potansiyel olarak büyük ve karmaşık arama alanlarının hesaplamalı bir aramasını kullanarak optimizasyon problemlerini çözmek için kullanılabilir (örneğin, arama algoritmaları kullanılarak). Örnekler arasında test senaryoları üretmek (Bkz. Bölüm 11.3), belirli bir kapsama kriterini karşılayan en küçük test senaryoları sayısını belirlemek ve regresyon test senaryolarını optimize etmek (Bkz. Bölüm 11.4) bulunmaktadır.

Yukarıdaki kategorizasyon, Yapay Zekâ tarafından gerçekleştirilebilen test görevleri ile farklı Yapay Zekâ teknolojileri arasında önemli bir örtüşme olduğu için zorunlu olarak geniş kapsamlıdır. Ayrıca bu sadece bir kategorizasyondur ve eşit derecede geçerli olabilecek diğerleri de oluşturulabilir.

11.1.1 Uygulamalı Egzersiz: Yapay Zekâ'nın Testlerde Kullanımı

Bir tartışma parçası olarak, öğrenciler şu anda Yapay Zekâ olarak uygulanması pratik olmayan test etkinliklerini ve görevlerini belirleyeceklerdir. Bunlar arasında şunlar olabilir:

- Test doğrulayıcılarını belirleme
- Belirsizlikleri açıklığa kavuşturmak ve eksik bilgileri almak için paydaşlarla iletişim kurma
- Kullanıcı deneyimine yönelik iyileştirmeler önerme
- Paydaş varsayımlarına meydan okumak ve rahatsız edici sorular sorma
- Kullanıcı ihtiyaçlarını anlama

Sınırlı görevler için kullanılacak zayıf Yapay Zekâ ile şu anda mevcut olmayan genel Yapay Zekâ arasında bir ayırım çizilmelidir. (Bkz. Bölüm 1.2).

11.2 Raporlanan Hataları Analiz Etmek için YZ Kullanımı

Bildirilen hatalar genellikle kategorize edilir, önceliklendirilir ve kopyaları tanımlanır. Bu etkinlik genellikle hata önceliklendirme veya analiz olarak adlandırılır ve hata çözümünde harcanan süreyi optimize etmeyi amaçlar. Yapay Zekâ, çeşitli yollarla bu aktiviteyi desteklemek için kullanılabilir, örneğin:

- Kategorizasyon: NLP [B25], hata raporlarındaki metni analiz etmek ve ardından diğer meta verilerle birlikte kümelenme algoritmalarına, örneğin k-en yakın komşular veya destek vektör makinelerine sunulabilecek, etkilenen fonksiyonelliğin alanı gibi konuları çıkarmak için kullanılabilir.

Bu algoritmalar, uygun hata kategorilerini belirleyebilir ve benzer veya yinelenen hataları vurgulayabilir. Yapay Zekâ tabanlı kategorizasyon, özellikle otomatik hata raporlama sistemleri (örneğin, Microsoft Windows ve Firefox için) ve birçok yazılım mühendisinin bulunduğu büyük projelerde faydalıdır.

- Kritiklik: En kritik hataların özelliklerine göre eğitilmiş Makine Öğrenimi modelleri, bildirilen hataların büyük bir yüzdesini oluşturan sistem arızalarına neden olma olasılığı en yüksek olan hataları belirlemek için kullanılabilir [B26].
- Atama: Makine Öğrenimi modelleri, hata içeriğine ve önceki geliştirici atamalarına dayanarak, belirli hataları düzeltmek için en uygun geliştiricileri önermektedir.

11.3 Test Senaryosu Oluşturmak için YZ Kullanımı

Testleri üretmek için Yapay Zekâ kullanımı, test varlıklarını hızlı bir şekilde oluşturmak ve kapsamı (örneğin, kod veya gereksinimler kapsamı) maksimize etmek için çok etkili bir teknik olabilir. Bu testlerin üretilmesi için temel, kaynak kod, kullanıcı arayüzü ve makine tarafından okunabilir bir test modelidir. Bazı araçlar ayrıca, sistem araçları veya günlük dosyaları aracılığıyla sistemin düşük seviyeli davranışlarının gözlemlenmesine dayanarak testler oluşturur [B27].

Ancak, testlerin temeli olarak gerekli davranışları tanımlayan bir test modeli kullanılmadıkça, bu tür test üretimi genellikle bir test doğrulayıcı probleminde muzdariptir. Çünkü Yapay Zekâ tabanlı araç, belirli bir test veri seti için beklenen sonuçların ne olması gerektiğini bilmez. Bir çözüm, uygun bir sistem kullanılabilirse sırt sırta test kullanmaktır (Bkz. Bölüm 9.3). Alternatif olarak, testler "uygulama yanıt vermiyor" veya sistem çökmesi olmadığı beklenen sonuç ile veya diğer benzer basit başarısızlık göstergeleri ile çalıştırılabilir.

Yapay Zekâ tabanlı test üretim araçları ile benzer Yapay Zekâ olmayan bulanıklık test araçlarını karşılaştıran araştırmalar, Yapay Zekâ tabanlı araçların eşdeğer kapsam seviyelerine ulaşabileceğini ve ortalama olarak bir başarısızlığa neden olmak için gerekli adım sayısını ortalama 15.000 adımdan yaklaşık 100 adıma düşürürken daha fazla hata bulabildiğini göstermektedir. Bu, hata ayıklamayı çok daha kolay hale getirir. [B27].

11.4 Regresyon Test Gruplarının Optimizasyonu için YZ Kullanımı

Bir sisteme değişiklikler yapıldıkça, yeni testler oluşturulur, uygulanır ve bir regresyon test grubu adayı haline gelir. Regresyon test gruplarının aşırı büyümesini önlemek için, test senaryolarını seçmek, önceliklendirmek ve hatta artırmak suretiyle daha etkili ve verimli bir regresyon test grubu oluşturmak üzere sık sık optimizasyon yapılmalıdır.

Yapay Zekâ tabanlı bir araç, örneğin, önceki test sonuçlarından elde edilen bilgileri, ilişkili hataları ve yapılan son değişiklikleri, örneğin daha sık bozulan özellikleri ve son değişikliklerden etkilenen kodu çalıştıran testleri analiz ederek regresyon test grubunun optimizasyonunu gerçekleştirebilir.

Araştırmalar, bir regresyon test grubunun boyutunda %50'ye varan azalmaların, hala çoğu hatayı tespit ederken elde edilebileceğini [B28], ve sürekli entegrasyon testi için test yürütme süresinde %40'a varan azalmaların, önemli bir hata tespit azalması olmadan ulaşabileceğini göstermektedir [B29].

11.5 Hata Tahmini için YZ Kullanımı

Hata tahmini, bir hatanın var olup olmadığını, kaç hata bulunduğunu veya hataların bulunup bulunamayacağını tahmin etmek için kullanılabilir. Bu yetenek, kullanılan aracın sofistikasyonuna bağlıdır.

Sonuçlar genellikle testlerin önceliklendirilmesi için kullanılır (örneğin, daha fazla hatanın tahmin edildiği bileşenler için daha fazla test).

Hata tahmini tipik olarak kaynak kod metrikleri, süreç metrikleri ve/veya insan ve organizasyonel metriklerine dayanır. Göz önünde bulundurulması gereken çok sayıda potansiyel faktör olduğundan, bu faktörler ile hata olasılığı arasındaki ilişkiyi belirlemek insan yeteneklerinin ötesindedir. Sonuç olarak, genellikle Makine Öğrenimi kullanılan Yapay Zekâ tabanlı bir yaklaşım kullanmak bir zorunluluktur. Hata tahmini, benzer bir durumda önceki deneyimlere dayandığında en etkilidir (örneğin, aynı kod tabanı ve/veya aynı geliştiricilerle).

Makine Öğrenimi kullanarak hata tahmini, birkaç farklı durumda başarıyla kullanılmıştır (örneğin, [B30] ve [B31]). En iyi tahmin edicilerin, daha yaygın kullanılan kaynak kod metrikleri, örneğin kod satırları ve siklomatik (döngüsel) karmaşıklık yerine, insan ve organizasyonel ölçümler olduğu bulunmuştur [B32].

11.5.1 Uygulamalı Egzersiz: Bir Hata Tahmin Sistemi Kurma

Öğrenciler, uygun bir veri seti kullanacaklar (örneğin, kaynak kod ölçümleri ve karşılık gelen hata verileri içeren) ve basit bir hata tahmin modeli oluşturacaklar ve bu modeli, benzer kodlardan kaynak kod ölçümleri kullanarak hataların olasılığını tahmin etmek için kullanacaklar.

Model, veri setinden en az dört özellik kullanmalı ve sınıf, seçilen özelliklere bağlı olarak sonuçların nasıl değiştiğini vurgulamak için birkaç farklı özellik kullanarak sonuçları incelemelidir.

11.6 Kullanıcı Arayüzlerini Test Etmek için YZ Kullanımı

11.6.1 Grafik Kullanıcı Arayüzünü (GUI) YZ ile Test Etme

Grafik Kullanıcı Arayüzü (GUI) üzerinden test etme, manuel testin (bileşen testi dışında) tipik yaklaşımıdır ve genellikle test otomasyonu girişimlerinin başlangıç noktasıdır. Sonuçlanan testler, test nesnesiyle insan etkileşimini taklit eder. Bu senaryolu test otomasyonu, kullanıcı arayüzü elementlerinin gerçek koordinatlarını veya arayüzün yazılım tanımlı nesne/widget'larını kullanarak, bir kaydet/oyunat yaklaşımı uygulayarak gerçekleştirilebilir. Ancak, bu yaklaşım nesne tanımlama ile ilgili birkaç dezavantajla karşı karşıyadır, bunlar arayüz değişikliklerine, kod değişikliklerine ve platform değişikliklerine duyarlılığı içerir.

Yapay Zekâ, çeşitli kriterler (örneğin, XPath, etiket, id, sınıf, X/Y koordinatları) kullanarak doğru nesnelere tanımlamak ve tarihsel olarak en kararlı tanımlama kriterlerini seçmek için Yapay Zekâ tabanlı araçlar kullanılarak bu yaklaşımın kırılabilirliğini azaltmak için kullanılabilir. Örneğin, uygulamanın belirli bir alanındaki bir butonun ID'si her sürümde değişebilir ve bu nedenle Yapay Zekâ tabanlı araç zamanla bu ID'ye daha az önem atayabilir ve diğer kriterlere daha fazla güvenebilir. Bu yaklaşım, kullanıcı arayüzündeki nesnelere testle eşleşen veya eşleşmeyen olarak sınıflandırır.

Alternatif olarak, görsel test etme, Grafik Kullanıcı Arayüzü (GUI) nesneleriyle gerçek bir kullanıcı arayüzü aracılığıyla aynı şekilde etkileşim kurmak için görüntü tanımayı kullanır ve bu nedenle alttaki kod ve arayüz tanımlarına erişmeye ihtiyaç duymaz. Bu, onu tamamen müdahalesiz ve altta yatan teknolojiye bağımsız kılar. Senaryoların sadece görünür kullanıcı arayüzü aracılığıyla çalışması gerekir. Bu yaklaşım, test uzmanının ekran

üzerindeki görüntülerle, butonlarla ve metin alanlarıyla, bir insan kullanıcının yaptığı gibi doğrudan etkileşimde bulunan senaryoları oluşturmasını sağlar, genel ekran düzeninden etkilenmeden test otomasyonunda görüntü tanıma kullanımı, gereken bilgi işlem kaynakları tarafından sınırlanabilir. Ancak, sofistike görüntü tanımayı destekleyen uygun fiyatlı YZ'nın bulunabilirliği artık bu yaklaşımı ana akım kullanım için mümkün kılmaktadır.

11.6.2 Grafik Kullanıcı Arayüzü (GUI)'nü Test Etmek için YZ Kullanımı

Makine Öğrenimi modelleri, kullanıcı arayüzü ekranlarının kabul edilebilirliğini belirlemek için kullanılabilir (örneğin, sezgisel yöntemler ve gözetimli öğrenme kullanılarak). Bu modellere dayanan araçlar, yanlış şekilde oluşturulmuş elementleri tanımlayabilir, bazı nesnelere erişilemez veya tespit edilmesi zor olup olmadığını belirleyebilir ve Grafik Kullanıcı Arayüzü (GUI)'nün görsel görünüşüyle ilgili çeşitli diğer sorunları tespit edebilir.

Görüntü tanıma, bilgisayar görüşü algoritmalarının bir formu olmasına rağmen, Yapay Zekâ tabanlı diğer bilgisayar görüşü formları da (örneğin, ekran görüntüleri) layout, boyut, pozisyon, renk, font veya nesnelere diğer görünür özelliklerinde istenmeyen değişiklikleri tanımlamak için görüntüleri karşılaştırmak için kullanılabilir. Bu karşılaştırmaların sonuçları, test nesnesine yapılan değişikliklerin kullanıcı arayüzünü olumsuz etkilemediğini kontrol etmek için regresyon testini desteklemek için kullanılabilir.

Ekranların kabul edilebilirliğini kontrol etme teknolojisi, tespit edilen kullanıcı arayüzü değişikliklerinin kullanıcılar tarafından kabul edilip edilmeyeceğini tavsiye edebilen veya bu değişikliklerin insan tarafından kontrol edilmesi gerekip gerekmediğini belirten daha sofistike Yapay Zekâ tabanlı regresyon test araçları oluşturmak için karşılaştırma araçları ile birleştirilebilir. Böyle Yapay Zekâ tabanlı araçlar, aynı uygulamanın farklı tarayıcılarda/cihazlarda/platformlarda doğru şekilde çalıştığını kontrol etmeyi amaçlayan farklı tarayıcılar, cihazlar veya platformlar için uyumluluk testini desteklemek için de kullanılabilir.

12. Referanslar

12.1 Standartlar

- [S01] ISO/IEC TR 29119-11:2020, Software and systems engineering — Software testing — Part 11 Guidelines on the testing of AI-based systems
- [S02] DIN SPEC 92001-1, Artificial Intelligence - Life Cycle Processes and Quality Requirements - Part 1: Quality Meta Model, <https://www.din.de/en/wdc-beuth:din21:303650673> (accessed May 2021).
- [S03] DIN SPEC 92001-2, Artificial Intelligence - Life Cycle Processes and Quality Requirements - Part 2: Technical and Organizational Requirements, <https://www.din.de/en/innovation-and-research/din-spec-en/projects/wdc-proj:din21:298702628> (accessed May 2021).
- [S04] ISO 26262 - <https://www.iso.org/standard/68383.html> (accessed May 2021)
- [S05] ISO/PAS 21448:2019, Road vehicles – Safety of the intended functionality (SOTIF) - <https://www.iso.org/standard/70939.html> (accessed May 2021)
- [S06] ISO/IEC 25010:2011, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 2011.
- [S07] ISO 26262-6:2018 - Road vehicles - Functional safety - Part 6: Product development at the software level.
- [S08] ISO/IEC/IEEE 29119-4:2015, Software and systems engineering — Software testing — Part 4: Test techniques.

12.2 ISTQB® Dokümanları

- [I01] ISTQB® Certified Tester Foundation Level Syllabus, Version 2018 V3.1 <https://www.istqb.org/downloads/category/2-foundation-level-documents.html> (accessed May 2021).
- [I02] ISTQB® Certified Tester Advanced Level Test Analyst Syllabus, Version 3.1, section 3.2.6 <https://www.istqb.org/downloads/category/75-advanced-level-test-analyst-v3-1.html> (accessed August 2021).
- [I03] ISTQB® Certified Tester AI Testing, Overview of Syllabus, Version 1.0

12.3 Kitaplar ve Makaleler

- [B01] Cadwalladr, Carole (2014). "Are the robots about to rise? Google's new director of engineering thinks so..." The Guardian. Guardian News and Media Limited., <https://www.theguardian.com/technology/2014/feb/22/robots-google-ray-kurzweil-terminator-singularity-artificial-intelligence> (accessed May 2021).
- [B02] Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 4th Edition, Pearson, 2020.
- [B03] M. Davies et al., "Advancing Neuromorphic Computing With Loihi: A Survey of Results and Outlook," Proceedings of the IEEE, vol. 109, no. 5, pp. 911–934, May 2021, doi: 10.1109/JPROC.2021.3067593.
- [B04] Chris Wiltz, Can Apple Use Its Latest AI Chip for More Than Photos?, Electronics & Test, Artificial Intelligence, <https://www.designnews.com/electronics-test/can-apple-use-its-latest-ai-chip-more-photos/153617253461497> (accessed May 2021).
- [B05] HUAWEI Reveals the Future of Mobile AI at IFA 2017, Huawei Press Release, <https://consumer.huawei.com/en/press/news/2017/ifa2017-kirin970/> (accessed May 2021).
- [B06] REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), April 2016, <https://eur-lex.europa.eu/eli/reg/2016/679/oj> (accessed May 2021)
- [B07] Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles J3016_201806, SAE, https://www.sae.org/standards/content/j3016_201806/ (accessed May 2021).
- [B08] G20 Ministerial Statement on Trade and Digital Economy: Annex. Available from: <https://www.mofa.go.jp/files/000486596.pdf> (accessed May 2021).
- [B09] Concrete Problems in AI Safety, Dario Amodei (Google Brain), Chris Olah (Google Brain), Jacob Steinhardt (Stanford University), Paul Christiano (UC Berkeley), John Schulman (OpenAI), Dan Man' e (Google Brain), March 2016. <https://arxiv.org/pdf/1606.06565> (accessed May 2021).
- [B10] Explainable AI: the basics, Policy briefing, Issued: November 2019 DES6051, ISBN: 978-1-78252-433-5, The Royal Society.
- [B11] The Ultimate Guide to Data Labeling for Machine Learning, www.cloudfactory.com/data-labeling-guide (accessed May 2021).
- [B12] Pei et al, DeepXplore: Automated Whitebox Testing of Deep Learning Systems, Proceedings of ACM Symposium on Operating Systems Principles (SOSP '17), Jan 2017.
- [B13] Sun et al, Testing Deep Neural Networks, https://www.researchgate.net/publication/323747173_Testing_Deep_Neural_Networks, accessed Nov 2019 (accessed May 2021).

- [B14] A. Odena and I. Goodfellow, TensorFuzz: Debugging Neural Networks with Coverage-Guided Fuzzing, ArXiv e-prints, Jul. 2018, <https://arxiv.org/pdf/1807.10875> (accessed May 2021)
- [B15] Riccio, V. et al, Testing Machine Learning based Systems: A Systematic Mapping. Empirical Software Engineering, <https://link.springer.com/article/10.1007/s10664-020-09881-0> (accessed May 2021)
- [B16] Baudel, Thomas et al, Addressing Cognitive Biases in Augmented Business Decision Systems., <https://arxiv.org/abs/2009.08127> (accessed May 2021)
- [B17] Papernot, N. et al, Transferability in machine learning: from phenomena to black-box attacks using adversarial samples, arXiv preprint arXiv:1605.07277, 2016. <https://arxiv.org/pdf/1605.07277> (accessed May 2021).
- [B18] Chen et al, Metamorphic Testing: A Review of Challenges and Opportunities, ACM Comput. Surv. 51, 1, Article 4, January 2018. https://www.researchgate.net/publication/322261865_Metamorphic_Testing_A_Review_of_Challenges_and_Opportunities (accessed May 2021).
- [B19] Huai Liu, Fei-Ching Kuo, Dave Towey, and Tsong Yueh Chen., How effectively does metamorphic testing alleviate the oracle problem?, IEEE Transactions on Software Engineering 40, 1, 4–22, 2014
- [B20] James Whittaker, Exploratory Software Testing: Tips, Tricks, Tours and Techniques to Guide Test Design, 1. Edition, Addison-Wesley Professional, 2009.
- [B21] L. Wilkinson, A. Anand, and R. Grossman. High-dimensional visual analytics: Interactive exploration guided by pairwise views of point distributions. Visualization and Computer Graphics, IEEE Transactions on, 12(6):1363–1372, 2006, <https://www.cs.uic.edu/~wilkinson/Publications/sorting.pdf> (accessed May 2021).
- [B22] Ryan Hafen and Terence Critchlow, EDA and MÖ – A Perfect Pair for Large-Scale Data Analysis, IEEE 27th International Symposium on Parallel and Distributed Processing, 2013, <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6651091> (accessed May 2021).
- [B23] Breck, Eric, Shanqing Cai, Eric Nielsen, Michael Salib, and D. Sculley., The MÖ Test Score: A Rubric for MÖ Production Readiness and Technical Debt Reduction, IEEE International Conference on Big Data (Big Data), 2017, <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8258038> (accessed May 2021).
- [B24] Harman, The Role of Artificial Intelligence in Software Engineering, In First International Workshop on Realizing AI Synergies in Software Engineering (RAISE), pp. 1-6. IEEE, June 2012, <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6227961> (accessed May 2021).
- [B25] Nilambri et al, A Survey on Automated Duplicate Detection in a Bug Repository, International Journal of Engineering Research & Technology (IJERT), 2014, <https://www.ijert.org/research/a-survey-on-automated-duplicate-detection-in-a-bug->

[repository-IJERTV3IS041769.pdf](#) (accessed May 2021)

- [B26] Kim, D.; Wang, X.; Kim, S.; Zeller, A.; Cheung, S.C.; Park, S. (2011). "Which Crashes Should I Fix First? Predicting Top Crashes at an Early Stage to Prioritize Debugging Efforts," in the IEEE Transactions on Software Engineering, volume 37, <https://ieeexplore.ieee.org/document/5711013> (accessed May 2021).
- [B27] Mao et al, Sapienz: multi-objective automated testing for Android applications, Proceedings of the 25th International Symposium on Software Testing and Analysis, July 2016, http://www0.cs.ucl.ac.uk/staff/K.Mao/archive/p_issta16_sapienz.pdf (accessed May 2021).
- [B28] Rai et al, Regression Test Case Optimization Using Honey Bee Mating Optimization Algorithm with Fuzzy Rule Base, World Applied Sciences Journal 31 (4): 654-662, 2014, https://www.researchgate.net/publication/336133351_Regression_Test_Case_Optimization_Using_Honey_Bee_Mating_Optimization_Algorithm_with_Fuzzy_Rule_Base (accessed May 2021).
- [B29] Dusica Marijan, Arnaud Gotlieb, Marius Liaaen. A learning algorithm for optimizing continuous integration development and testing practice, Journal of Software : Practice and Experience, Nov 2018.
- [B30] Tosun et al, AI-Based Software Defect Predictors: Applications and Benefits in a Case Study, Proceedings of the Twenty-Second Innovative Applications of Artificial Intelligence Conference (IAAI-10), 2010.
- [B31] Kim et al, Predicting Faults from Cached History, 29th International Conference on Software Engineering (ICSE'07), 2007.
- [B32] Nagappan 2008 Nagappan et al, The Influence of Organizational Structure on Software Quality: An Empirical Case Study, Proceedings of the 30th international conference on Software engineering (ICSE'08), May 2008.
- [B33] Kuhn et al, Software Fault Interactions and Implications for Software Testing, IEEE Transactions on Software Engineering vol. 30, no. 6, (June 2004) pp. 418-421.

12.4 Diğer Referanslar

The following references point to information available on the Internet. Even though these references were checked at the time of publication, the ISTQB® cannot be held responsible if the references are no longer available.

- [R01] Wikipedia contributors, "AI effect," Wikipedia, https://en.wikipedia.org/wiki/AI_effect (accessed May 2021).
- [R02] <https://mxnet.apache.org/> (accessed May 2021).
- [R03] <https://docs.microsoft.com/en-us/cognitive-toolkit/> (accessed May 2021).
- [R04] IBM Watson, <https://www.ibm.com/watson/ai-services>

- [R05] <https://www.tensorflow.org/> (accessed May 2021).
- [R06] <https://keras.io/> (accessed May 2021).
- [R07] <https://pytorch.org/> (accessed May 2021).
- [R08] https://scikit-learn.org/stable/whats_new/v0.23.html (accessed May 2021).
- [R09] NVIDIA VOLTA, <https://www.nvidia.com/en-us/data-center/volta-gpu-architecture/> (accessed May 2021).
- [R10] Cloud TPU, <https://cloud.google.com/tpu/> (accessed May 2021).
- [R11] Edge TPU, <https://cloud.google.com/edge-tpu/> (accessed May 2021).
- [R12] Intel® Nervana™ Neural Network processors deliver the scale and efficiency demanded by deep learning model evolution, <https://www.intel.ai/nervana-nnp/> (accessed May 2021).
- [R13] The Evolution of EyeQ, <https://www.mobileye.com/our-technology/evolution-eyeq-chip/> (accessed May 2021).
- [R14] ImageNet - <http://www.image-net.org/> (accessed May 2021).
- [R15] Google's BERT - <https://github.com/google-research/bert> (accessed May 2021).
- [R16] <https://www.kaggle.com/datasets> (accessed May 2021).
- [R17] <https://www.kaggle.com/paultimothymooney/2018-kaggle-machine-learning-data-science-survey> (accessed May 2021).
- [R18] MÖCommons - <https://mlcommons.org/> (accessed May 2021).
- [R19] DAWNbench – <https://dawn.cs.stanford.edu/benchmark> (accessed May 2021).
- [R20] MÖMark – <https://www.eembc.org/mlmark> (accessed May 2021).
- [R21] <https://digital-strategy.ec.europa.eu/en/library/assessment-list-trustworthy-artificial-intelligence-altai-self-assessment> Shaping Europe's digital future (europa.eu) (accessed August 2021)
- [R22] <https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai> (accessed August 2021)
- [R23] Google GraphicsFuzz, <https://github.com/google/graphicsfuzz> (accessed May 2021).
- [R24] <http://www.openrobots.org/morse/doc/0.2.1/morse.html> (accessed May 2021).
- [R25] <https://ai.facebook.com/blog/open-sourcing-ai-habitat-a-simulation-platform-for-embodied-ai-research/> (accessed May 2021).
- [R26] <https://www.nvidia.com/en-gb/self-driving-cars/drive-constellation/> (accessed May 2021).

[R27]

<https://uk.mathworks.com/discovery/artificial-intelligence.html#ai-with-matlab> (accessed May 2021)

13. Ek A – Kısaltmalar

Kısaltma	Açıklama
AI	Yapay Zekâ
AlaaS	Hizmet Olarak Yapay Zekâ
API	Uygulama Programlama Arayüzü
AUC	Eğri Altındaki Alan
DL	Derin Öğrenme
DNN	Derin Sinir Ağı
EDA	Keşifsel Veri Ağı
EU	Avrupa Birliği
FN	Yanlış Negatif
FP	Yanlış Pozitif
GDPR	Genel Veri Koruma Yönetmeliği
GPU	Grafik İşleme Birimi
GUI	Grafiksel Kullanıcı Arayüzü
LIME	Yerel Yorumlanabilir Model-Agnostik Açıklamalar
MC/DC	Modifiye Koşul Karar Kapsamı
MÖ	Makine Öğrenimi
MR	Metamorfik İlişki
MSE	Ortalama Kare Hatası
MT	Metamorfik Test
NLP	Doğal Dil İşleme
ROC	Alıcı İşletim Karakteristiği
SUT	Test Edilen Sistem
SVM	Destek Vektör Makinesi
TN	Doğru Negatif
TP	Doğru Pozitif
XAI	Açıklanabilir Yapay Zekâ

14. Ek B – Yapay Zekâ Özellikleri ve Diğer Terimler

Terim Adı	Tanım
accuracy (doğruluk)	Bir sınıflandırıcıyı değerlendirmek için kullanılan MÖ (Makine Öğrenimi) işlevsel performans metriği, doğru tahminlerin oranını ölçer (ISO/IEC TR 29119-11'e göre)
activation function (aktivasyon fonksiyonu)	Bir sinir ağındaki bir nöronla ilişkilendirilen ve nöronun girdilerinden nöronun çıktısını belirleyen formül
activation value (aktivasyon değeri)	Bir sinir ağındaki bir nöronun aktivasyon fonksiyonunun çıktısı
adversarial attack (karşıt saldırı)	Bir MÖ modelinin başarısız olmasına neden olmak için kasten kullanılan adversaryel örnekler
AI as a Service (AlaaS) (Hizmet olarak Yapay Zekâ)	Yapay zekâ ve Yapay Zekâ geliştirme hizmetlerinin merkezi olarak barındırıldığı bir yazılım lisanslama ve teslim modeli
AI component (YZ bileşeni)	Yapay zekâ işlevselliği sağlayan bir bileşen
AI effect (YZ etkisi)	Teknoloji ilerledikçe daha önce Yapay Zekâ olarak etiketlenmiş bir sistemin artık Yapay Zekâ olarak kabul edilmemesi durumu (ISO/IEC TR 29119-11)
AI-based system (YZ tabanlı sistem)	Bir veya daha fazla Yapay Zekâ bileşenini entegre eden bir sistem
AI-specific processor (Yapay zekâya özgü işlemci)	Yapay Zekâ uygulamalarını hızlandırmak için tasarlanmış özel bir donanım türü
algorithmic bias (algoritmik yanlılık)	MÖ algoritması tarafından neden olunan bir yanlılık türü
annotation (etiketleme)	Görsellerdeki nesnelere sınıflandırma için etiketlenmiş veriler sağlamak üzere sınırlayıcı kutularla tanımlama etkinliği
area under curve (AUC) (eğri altındaki alan)	Bir sınıflandırıcının iki sınıfı ne kadar iyi ayırt edebildiğini ölçen bir metrik
artificial intelligence (AI) (yapay zekâ)	Mühendislik ürünü bir sistemin bilgi ve becerileri edinme, işleme, oluşturma ve uygulama yeteneği (ISO/IEC TR 29119-11)
association (çağırışım)	Örnekler arasındaki ilişkileri ve bağımlılıkları belirleyen gözetimsiz öğrenme tekniği
augmentation (arttırma)	Mevcut bir veri setine dayanarak yeni veri noktaları oluşturma etkinliği

Terim Adı	Tanım
automation bias (otomasyon yanlılığı) synonym: complacency bias	Bir kişinin otomatik karar verme sisteminin önerilerini diğer kaynaklara (kendisi dahil) tercih etmesi sonucu oluşan yanlılık türü eşanlamı: kayıtsız yanlılık
autonomous system (otonom sistem)	İnsan müdahalesi olmadan uzun süre çalışabilen bir sistem
autonomy (otonomi)	Bir sistemin insan müdahalesi olmadan uzun süre çalışabilme yeteneği (ISO/IEC TR 29119-11)
Bayesian model (Bayes modeli)	Model girişlerinin ve çıkışlarının belirsizliğini temsil etmek için olasılığı kullanan bir istatistiksel model
Bayesian technique (Bayes tekniği)	İstatistiksel bir modelin parametreleri olarak öncesi ve sonrası olasılık dağılımlarını ele alan teknik
bias (yanlılık)	Belirli nesnelere, kişilerin veya grupların diğerlerine kıyasla sistematik olarak farklı muamele görmesi (ISO/IEC DIS 22989)
big data (büyük veri)	Hacim, çeşitlilik, hız ve/veya değişkenlik açısından özellikleri nedeniyle özel teknolojiler ve teknikler gerektiren büyük veri setleri
case-based reasoning (vakaya dayalı akıl yürütme)	Geçmişte benzer sorunların çözümlerine dayanarak yeni bir sorunu çözme tekniği
chatbot (sohbet robotu)	Metin veya metinden konuşmaya dönüştürme yoluyla sohbet yürüten bir uygulama
classification (sınıflandırma)	Verilen bir girdi için çıktı sınıfını tahmin eden bir MÖ işlevi (ISO/IEC TR 29119-11'e göre)
classifier (sınıflandırıcı)	Sınıflandırma için kullanılan bir MÖ modeli
clustering (kümeleme)	Benzer veri noktalarını bir araya getiren bir MÖ işlevi
clustering algorithm (kümeleme algoritması)	Benzer nesnelere kümeler halinde gruplamak için kullanılan bir MÖ algoritması
concept drift (kavram kayması)	Kullanıcı beklentilerindeki, davranışlarındaki ve operasyonel ortamdaki değişikliklerin MÖ modelinin tahminlerinin doğruluğunda zamanla algılanan değişiklik

Terim Adı	Tanım
confusion matrix (karışıklık matrisi)	Bir sınıflandırma algoritmasının MÖ işlevsel performansını özetlemek için kullanılan bir teknik
data acquisition (veri edinimi)	Bir MÖ modeli tarafından çözülecek iş problemine ilişkin verilerin toplanması etkinliği
data labelling (veri etiketleme)	MÖ'de sınıflandırmayı desteklemek için ham verilerdeki nesnelere anlamlı etiketler ekleme etkinliği
data pipeline (veri akışı)	Bir MÖ algoritması tarafından eğitimi veya bir MÖ modeli tarafından tahmini desteklemek için girdi verilerini sağlama amacıyla veri hazırlama etkinliklerinin uygulanması
data point (veri noktası)	Bir veri setinin parçası olarak kullanılan tek bir gözlemi içeren bir veya daha fazla ölçüm kümesi
data poisoning (veri zehirlenmesi)	Bir MÖ modeline yönelik eğitim veya girdi verilerinin kasten ve kötü niyetli manipülasyonu
data preparation (veri hazırlığı)	MÖ iş akışında veri edinme, veri ön işleme ve özellik mühendisliği etkinlikleri
data pre-processing (veri ön işleme)	MÖ iş akışında veri temizleme, veri dönüşümü, veri artırma ve veri örnekleme etkinlikleri
data visualization (veri görselleştirme)	Verilerin ilişkilerini, eğilimlerini ve kalıplarını grafiksel olarak temsil etme tekniği
dataset (veri seti)	MÖ'de eğitim, değerlendirme, test ve tahmin için kullanılan bir veri koleksiyonu
decision threshold (karar eşiği)	Bir tahmin fonksiyonunun sonucunu belirli bir değerin üzerinde veya altında ikili bir sonuca dönüştüren bir değer
decision tree (karar ağacı)	Düğümüleri kararları, dalları ise olası sonuçları temsil eden ağaç benzeri bir MÖ modeli
deductive classifier (tümdengelimli sınıflandırıcı)	Girdi verilerine çıkarım ve mantık uygulamasına dayanan bir sınıflandırıcı
deep learning (DL) (derin öğrenme)	Çok katmanlı sinir ağları kullanarak yapılan MÖ
deep neural network (derin nöral ağ)	Birkaç katmanlı nöronlardan oluşan bir sinir ağı

Terim Adı	Tanım
defect prediction (hata tahminleme)	Test nesnesi içindeki hataların meydana geleceği alanları veya mevcut hata miktarını tahmin etme tekniği
deterministic system (belirsiz sistem)	Belirli bir girdi seti ve başlangıç durumundan aynı çıktı setini ve son durumu üretecek bir sistem
edge computing (kenar hesaplama)	Bilgi işleminin bu bilginin kullanıldığı yere yakın gerçekleştirildiği dağıtık bir mimarinin parçası
epoch (dönem)	Tüm eğitim veri seti üzerinde MÖ eğitiminin bir yinelemesi
evolution (evrim)	Daha düşük, daha basit veya daha kötü bir durumdan daha yüksek, daha karmaşık veya daha iyi bir duruma sürekli değişim süreci
expert system (uzman sistem)	İnsan uzmanlığından geliştirilen bir bilgi tabanından çıkarım yaparak belirli bir alan veya uygulama alanında sorunları çözmek için kullanılan bir Yapay Zekâ tabanlı sistem
explainability (açıklanabilirlik)	Yapay zekâ tabanlı sistemin belirli bir sonuca nasıl ulaştığının anlaşılma düzeyi (ISO/IEC TR 29119-11)
explainable AI (XAI) (açıklanabilir yapay zekâ)	Yapay zekâ sistem çıktılarının etkileyen faktörleri anlama ile ilgili çalışma alanı
exploratory data analysis (EDA) (keşfedici veri analizi)	Özellik mühendisliğini desteklemek için kullanılan veri etkileşimli, hipotez odaklı ve görsel keşfi
F1-Score (F1 skoru)	Bir sınıflandırıcıyı değerlendirmek için kullanılan, hatırlama ve kesinlik arasında denge sağlayan bir MÖ işlevsel performans metriği
false negative (FN) (yanlış negatif)	Modelin yanlışlıkla negatif sınıfı tahmin ettiği bir MÖ model tahmini
false positive (FP) (yanlış pozitif)	Modelin yanlışlıkla pozitif sınıfı tahmin ettiği bir MÖ model tahmini
feature (özellik)	Bir MÖ algoritması tarafından eğitim için ve bir MÖ modeli tarafından tahmin için kullanılan giriş verilerinin bireysel ölçülebilir bir niteliği
feature engineering (özellik mühendisliği)	MÖ modelinde ortaya çıkması gereken temel ilişkileri en iyi temsil eden ham verilerdeki öznitelikleri eğitim verilerinde kullanmak için belirleme etkinliği (ISO/IEC TR 29119-11)

Terim Adı	Tanım
flexibility (esneklik)	Bir sistemin başlangıçta belirtilen bağlamlar dışında çalışabilme yeteneği (ISO/IEC TR 29119-11'e göre)
fuzzy logic (bulanık mantık)	Kısmi doğruluk kavramına dayanan ve kesinlik faktörleri 0 ile 1 arasında temsil edilen bir mantık türü
general AI (genel yapay zekâ)	Tüm bilişsel yetenekler boyunca insanla karşılaştırılabilir zeki davranış sergileyen Yapay Zekâ (ISO/IEC TR 29119-11)
Synonym: strong AI	Eşanlamlı: güçlü yapay zekâ
General Data Protection Regulation (GDPR) (Genel Veri Koruma Yönetmeliği)	Avrupa Birliği (AB) vatandaşlarının ve Avrupa Ekonomik Alanı'ndaki vatandaşların verileri için geçerli olan veri koruma ve gizlilik düzenlemesi
graphical processing unit (GPU) (grafik işlem birimi)	Bir çerçeve arabelleğindeki görüntülerin oluşturulmasını hızlandırmak için bellek manipülasyonu ve değiştirilmesi amacıyla tasarlanmış uygulamaya özel entegre devre
ground truth (asıl gerçek)	Gerçek veya doğru olduğu bilinen doğrudan gözlem ve ölçümle sağlanan bilgi
Hyperparameter (hiperparametre)	Bir MÖ modelinin eğitimini kontrol etmek veya bir MÖ modelinin yapılandırmasını ayarlamak için kullanılan parametre
hyperparameter tuning (hiperparametre ayarlaması)	Belirli hedeflere dayalı olarak optimal hiperparametreleri belirleme etkinliği
inappropriate bias (Uygunsuz yanlılık)	Belirli bir grup için olumsuz sonuçlara yol açan bir sistem yanlılığı türü
intelligent agent (akıllı ajan)	Hedeflerine ulaşmak için gözlem ve eylemleri kullanan otonom bir program
inter-cluster metric (kümeler arası metrikler)	Farklı kümelerdeki veri noktalarının benzerliğini ölçen bir metrik
interpretability (yorumlanabilirlik)	Altta yatan yapay zekâ teknolojisinin nasıl çalıştığını anlama düzeyi (ISO/IEC TR 29119-11)
intra-cluster metric (küme içi metrikler)	Bir küme içindeki veri noktalarının benzerliğini ölçen bir metrik
k-nearest neighbor (k-en yakın komşu)	Bir veri noktasının grup üyeliğinin, ona en yakın veri noktalarının grup üyeliğine bağlı olarak tahmin edildiği bir sınıflandırma yaklaşımı

Terim Adı	Tanım
learning algorithm (öğrenme algoritması)	Eğitim veri setinin özelliklerine dayalı olarak bir MÖ modeli üreten program
LIME method (LIME metodu / Yerel Açıklanabilir Model-Agnostik Açıklamalar metodu)	Bir MÖ modelinin tahminlerini açıklamak için kullanılan Yerel Açıklanabilir Model-Agnostik Açıklamalar programı
linear regression (doğrusal regresyon)	Hedef değişkenin sayısal olduğu durumlarda, gözlemlenen verilere lineer bir denklem uydurarak değişkenler arasındaki ilişkiyi modelleyen istatistiksel teknik
logistic regression (lojistik regresyon)	Hedef değişkenin sayısal değil kategorik olduğu durumlarda değişkenler arasındaki ilişkiyi modelleyen istatistiksel teknik
machine learning (ML) (makine öğrenimi)	Sistemlerin verilerden veya deneyimlerden öğrenmesini sağlayan hesaplama tekniklerini kullanma süreci (ISO/IEC TR 29119-11)
mean square error (MSE) (ortalama kare hatası)	Tahmini değerler ile gerçek değerler arasındaki ortalama karesel farkı ölçen istatistiksel ölçüt
ML algorithm (makine öğrenimi algoritması)	Eğitim veri setinden bir MÖ modeli oluşturmak için kullanılan algoritma
ML benchmark süite (MÖ için kıyaslama paketleri)	Bir dizi değerlendirme metriği üzerinden MÖ modellerini ve MÖ algoritmalarını karşılaştırmak için kullanılan veri seti
ML framework (MÖ çerçevesi)	Bir MÖ modelinin oluşturulmasını destekleyen bir araç veya kütüphane
ML function (MÖ fonksiyonu)	Sınıflandırma, regresyon veya kümeleme gibi bir MÖ modeli tarafından uygulanan işlevsellik
ML model evaluation (MÖ model değerlendirmesi)	Elde edilen MÖ işlevsel performans metriklerinin, gerekli kriterler ve diğer MÖ modellerinin kriterleri ile karşılaştırılması süreci
ML model training (MÖ model eğitimi)	Bir MÖ algoritmasının eğitim veri setine uygulanarak bir MÖ modeli oluşturma süreci
ML model tuning (MÖ model ayarlaması)	Optimum performansı elde etmek için hiperparametrelerin test edilmesi süreci
ML system (MÖ sistemi)	Bir veya daha fazla MÖ modelini entegre eden bir sistem
ML workflow (makine öğrenimi iş akışı)	Bir MÖ modelinin geliştirilmesi ve dağıtımının yönetildiği etkinliklerin sıralı dizisi

Terim Adı	Tanım
multi-agent system (çoklu ajan sistemi) synonym: weak AI	Birden fazla akıllı ajan içeren bir sistem eşanlamlı: zayıf yapay zekâ
narrow AI (dar yapay zekâ)	Belirli bir problemi çözmek için belirlenmiş tek bir göreve odaklanmış Yapay Zekâ (ISO/IEC TR 29119-11)
natural language processing (NLP) (doğal dil işleme)	Doğal dilleri okuyabilme, anlayabilme ve anlam çıkarabilme yeteneği sağlayan bir bilişim alanı
neural network (doğal ağ) synonym: artificial neural network	Ayarlanabilir ağırlıklara sahip bağlantılarla bağlanan ilkel işlem elemanlarından oluşan bir ağ; her eleman girdilerine doğrusal olmayan bir fonksiyon uygulayarak bir değer üretir ve bu değeri diğer elemanlara iletir veya çıktı olarak sunar (ISO/IEC 2382) eşanlamlı: yapay sinir ağı
neural network Trojan (sinir ağı truva atı)	Daha sonra kötüye kullanma amacıyla veri zehirlenme saldırısı kullanılarak bir sinir ağına enjekte edilen bir güvenlik açığı
neuromorphic processor (nöromorfik işlemci)	İnsan beynindeki biyolojik nöronları taklit etmek üzere tasarlanmış entegre devre
neuron (nöron)	Genellikle birden fazla giriş değeri alan ve bir aktivasyon değeri üreten bir sinir ağı düğümü
noise (gürültü)	Verideki bozulma veya bozukluk
non-deterministic system (belirsiz sistem)	Belirli bir girdi seti ve başlangıç durumu verildiğinde her zaman aynı çıktı setini ve son durumu üretmeyen bir sistem
outlier (aykırı değer)	Veri dağılımının genel deseninin dışında kalan bir gözlem
overfitting (aşırı uyum)	Eğitim veri setine çok yakın bir şekilde uyan, bu nedenle yeni verilere genelleme yapmakta zorlanan bir MÖ modeli oluşturma (ISO/IEC TR 29119-11'e göre)
perceptron (perseptron)	Sadece bir katmana ve bir nörona sahip bir sinir ağı
precision (hassasiyet)	Pozitif olarak tahmin edilenlerin ne kadarının doğru olduğunu ölçen bir MÖ işlevsel performans metriği (ISO/IEC TR 29119-11'e göre)

Terim Adı	Tanım
pre-trained model (eğitim öncesi model)	Alındığında zaten eğitilmiş olan bir MÖ modeli
probabilistic system (olasılıksal sistem)	Davranışı olasılıklar cinsinden tanımlanan; dolayısıyla çıktıları mükemmel şekilde tahmin edilemeyen bir sistem
procedural reasoning (prosedürel akıl yürütme)	Dinamik ortamlarda karmaşık görevleri yerine getirebilen gerçek zamanlı mantık sistemleri inşa etmek için kullanılan yapay zekâ teknolojisi
random forest (rastgele orman)	Sınıflandırma, regresyon ve diğer görevler için çok sayıda karar ağacı oluşturup çalıştırarak ve ardından ya sınıfın modunu ya da tek tek ağaçların ortalama tahminini çıkararak çalışan makine öğrenimi teknolojisi
reasoning technique (akıl yürütme teknikleri)	Mevcut bilgiden mantıksal teknikler kullanarak sonuçlar çıkaran yapay zekâ (ISO/IEC TR 29119-11'e göre)
recall (hatırlama)	Gerçek pozitiflerin ne kadarının doğru tahmin edildiğini ölçen bir MÖ işlevsel performans metriği (ISO/IEC TR 29119-11'e göre)
synonym: sensitivity	Eşanlamlı: duyarlılık
receiver operating characteristic (ROC) curve (Alıcı İşletim Karakteristiği eğrisi)	Bir ikili sınıflandırıcının ayırt etme eşiği değiştikçe yeteneğini gösteren grafiksel çizim
regression (regresyon)	Verilen bir giriş için sayısal veya sürekli bir çıktı değeri üreten bir MÖ işlevi (ISO/IEC TR 29119-11'e göre)
regression model (regresyon modeli)	Verilen sayısal giriş için beklenen çıktısı sürekli bir değişken olan bir MÖ modeli (ISO/IEC DIS 23053'e göre)
reinforcement learning (takviyeli öğrenme)	Bir amacı gerçekleştirmek için deneme ve ödül süreci kullanarak bir MÖ modeli inşa etme etkinliği (ISO/IEC TR 29119-11'e göre)
reward function (ödül fonksiyonu)	Pekiştirmeli öğrenmenin başarısını tanımlayan bir fonksiyon
reward hacking (ödül hileciliği)	Asıl amacı karşılamaya zarar verecek şekilde ödül fonksiyonunu en üst düzeye çıkarmak için bir akıllı ajan tarafından gerçekleştirilen etkinlik (ISO/IEC TR 29119-11'e göre)
R-squared (R-kare)	Veri noktalarının uydurulan regresyon çizgisine ne kadar yakın olduğunu ölçen istatistiksel ölçü
Synonym: coefficient of determination	Eşanlamlı: belirlilik katsayısı

Terim Adı	Tanım
rule engine (kural motoru)	Belirli koşullar sağlandığında hangi eylemlerin gerçekleşmesi gerektiğini belirleyen bir dizi kural
safety (güvenlik)	Bir sistemin, tanımlanmış koşullar altında, insan hayatını, sağlığını, malı veya çevreyi tehlikeye atmayan bir duruma yol açmadığı beklentisi (ISO/IEC/IEEE 12207)
sample bias (örneklem yanlılık)	MÖ'nin uygulandığı veri alanını tam olarak temsil etmeyen veri setinden kaynaklanan bir yanlılık türü
search algorithm (arama algoritması)	Hedef durum veya yapı ulaşıncaya kadar tüm olası durumların veya yapıların bir alt kümesini sistematik olarak ziyaret eden algoritma (ISO/IEC TR 29119-11'e göre)
self-learning system (kendi kendine öğrenme sistemi)	Deneme-yanılma yoluyla öğrenmeye dayalı olarak davranışını değiştiren uyarlanabilir bir sistem (ISO/IEC TR 29119-11'den sonra)
silhouette coefficient (siluet katsayısı)	Ortalama küme içi ve küme dışı farklılıklara dayalı olarak -1 ile +1 arasında bir kümeleme ölçüsü eşanlamlı: siluet skoru
synonym: silhouette score	
super AI (süper YZ)	İnsan yeteneklerini çok aşan yapay zekâ tabanlı bir sistem
supervised learning (denetimli öğrenme)	Etiketli bir veri seti kullanarak bir MÖ modelinin eğitilmesi
support vector machine (SVM) (destek vektör makinası)	Veri noktalarının çok boyutlu uzayda vektörler olarak görüldüğü ve bir hiper düzlemlerle ayrıldığı bir MÖ tekniği
technological singularity (teknolojik tekillik)	Teknolojik ilerlemelerin artık insanlar tarafından kontrol edilemediği gelecekteki bir nokta. (ISO/IEC TR 29119-11'e göre)
test oracle problem (sözde test sonucunu bilen problemi)	Belirli bir test girdisi ve durum için bir testin geçip geçmediğini belirleme zorluğu
training dataset (eğitim veri seti)	Bir MÖ modelini eğitmek için kullanılan veri seti
transfer learning (transfer öğrenme)	Eğitilmiş bir MÖ modelini farklı bir ilgili görevi yerine getirecek şekilde değiştirme tekniği
transparency (şeffaflık)	YZ tabanlı sistemin kullandığı algoritma ve verilerin görünürlük düzeyi (ISO/IEC TR 29119-11'e göre)

Terim Adı	Tanım
true negative (TN) (doğru negatif)	Modelin negatif sınıfı doğru tahmin ettiği bir tahmin
true positive (TP) (doğru pozitif)	Modelin pozitif sınıfı doğru tahmin ettiği bir tahmin
underfitting (yetersiz uyum)	Eğitim veri setinin altındaki eğilimi yansıtmayan, bu nedenle doğru tahminler yapmakta zorlanan bir MÖ modeli oluşturma (ISO/IEC TR 29119-11)
unsupervised learning (denetimsiz öğrenme)	Etiketlenmemiş bir veri seti kullanarak bir MÖ modelinin eğitilmesi
validation dataset (doğrulama veri seti)	Bir MÖ modelini ayarlamak amacıyla, eğitilmiş bir MÖ modelini değerlendirmek için kullanılan veri seti
von Neumann architecture (von Neumann mimarisi)	Bellek, merkezi işlem birimi, kontrol birimi, giriş ve çıkış olmak üzere beş ana bileşenden oluşan bir bilgisayar mimarisi
weight (ağırlık)	Bir sinir ağındaki nöronlar arasındaki bağlantının içsel değişkeni; çıktılarının nasıl hesaplandığını etkiler ve sinir ağı eğitildikçe değişir

15. Dizin

A/B testi 69, 72, 77	şifreleme 56
kabul testleri 55	hata tahmini 84
uyarlanabilirlik 24, 54, 57	deneyime dayalı test 59, 60
karşıt saldırı 70	açıklanabilirlik 27, 65, 79
karşıt örnek 38, 70	keşif testi 74,76
API testi 55	esneklik 24
saldırgan 22, 42, 70	takip test senaryosu 72, 73
otomasyon yanlılığı 56	fonksiyonel uygunluk 66
uygunluk 14, 56, 66	bulanıklık testi 83
sırt sırta test 69, 71, 77	grafik kullanıcı arayüzü 84, 85
yanlılık 25	giriş verisi testi 31
kara kutu testi 64, 65, 70	taşınabilirlik 57, 66
kontrol listesine dayalı test 74	entegrasyon testi 55, 57, 83
sınıflandırma 30, 46, 65	yorumlanabilirlik 26, 60, 65
kümeleme 29, 30, 46	makine öğrenimi 74, 77, 84
kombinasyonlu test 71	bakım kolaylığı 66
uyumluluk 66, 85	metamorfik ilişki 72, 73, 74
karmaşıklık 27, 84	metamorfik test 72, 73, 77
bileşen 20, 24, 25	MÖ algoritmaları 31, 32, 35
entegrasyon testi 55	MÖ işlevsel performans kriterleri 32, 55, 58
bileşen testi 55	MÖ işlevsel performans ölçütleri 31, 32
onay testi 58, 64	MÖ modeli 30, 38, 40
sürekli test 62	MÖ model testi 55
kapsam 49, 52, 71	MÖ iş akışı 34, 38
veri zehirlenmesi 70, 77	sinir ağı 19, 32, 50
veri gizliliği 41, 42	nöron kapsamı 52
hata ayıklama 83	işlevsel olmayan gereksinim 55, 57, 67
karar ağacı 32, 99	fonksiyonel olmayan test 73
dinamik test 54, 58, 65	operasyonel ortam 24, 79
etkinlik 52, 71, 82	
verimlilik 77	

ikili test 71, 77
performans verimliliği 39, 46, 66
zehirlenme saldırısı 70
taşınabilirlik 57, 66
sözde sonucu bilen 71, 72
kalite özellikleri 24, 35, 46
regresyon 18, 30, 46
regresyon testi 62, 77, 85
takviyeli öğrenme 30, 31, 35
güvenilirlik 66
ödül hileciliği 26, 27
ölçeklenebilir 19, 39, 80
güvenlik 22, 27, 42
güvenlik açıkları 42, 70
işaret değişimi kapsamı 52
işaret işaret kapsamı 52
simülatör 55, 80
kaynak test senaryosu 72, 73
denetimli öğrenme 30, 42, 43
sistem testi 55, 59, 71
test edilen sistem 56, 72, 79
test otomasyonu 84, 85
test verileri 56
test veri seti 32, 40, 57
test tasarımı 62
test ortamı 62, 79, 80
test seviyesi 55
test modeli 83
test sonucunu bilen 72, 73, 74
test planlama 79
test grubu 71, 83
test tekniği 71, 73, 77
test edilebilirlik 53, 65
eşik kapsamı 52
tur 74, 76
eğitim veri seti 20, 32, 36
şeffaflık 26, 27, 65
denetimsiz öğrenme 30, 35
kullanılabilirlik 57, 66, 60
kullanıcı deneyimi 82
kullanıcı arayüzü 82, 84, 85
doğrulama 32, 40
doğrulama veri seti 40
değer değişim kapsamı 49
sanal test ortamı 79, 80
görsel test 84
beyaz kutu testi 52, 57, 65