Bölüm 8

PYTHON SCRAPY KÜTÜPHANESİ İLE WEB'DEN BİLGİ TOPLAMA VE VERİ ÇEKME TEKNİKLERİ

Kâmil Abdullah EŞİDİR[1]

1. GİRİŞ

Günümüzde birçok alanda ve sektörde faaliyet gösteren işletmelerin işleyiş süreçlerini kontrol etmek ve verimliliği artırmak için bilgi teknolojilerinin kullanımı neredeyse zorunludur. Bilgi teknolojileri, işletmelerin daha verimli, hızlı ve rekabetçi olmalarını sağlayan önemli bir araçtır. İşletmeler, bu teknolojileri stratejik bir şekilde kullanarak büyümeye çalışır (Kılınç ve Aydın, 2019).

Web, İnternet üzerindeki milyonlarca sayfa ve içeriğin bulunduğu devasa bir veri deposudur. İnternet üzerinde yayınlanan bilgileri, metinleri, görselleri, videolari, ses dosyalarını ve diğer medya türlerini içerir. Web sayfaları, bu bilgilere erişim sağlayan ve bu bilgileri kullanıcılarla paylaşan yapıları ifade etmektedir. Web sayfaları, kullanıcıların bilgiye erişimini kolaylaştırır ve İnternet üzerindeki içeriğin keşfedilmesini sağlar. Ayrıca, web sayfaları sosyal medya, e-ticaret, eğitim, eğlence ve daha birçok alanda hizmet sunmaktadır (Alanoğlu ve Akcayol, 2023). Web, modern toplumun bilgiye erişiminde ve paylaşımında kritik bir rol oynamaktadır ve İnternet'in temel bileşenlerindendir.

Web sitelerinden veri çekme veya web kazıma (web scraping), günümüzde birçok sektör için büyük öneme sahip bir faaliyettir. Bu işlem, çeşitli alanlarda kullanılan oldukça değerli verileri elde etmek için kullanılmaktadır. Scrapy, web sitelerinden veri çekmeyi daha kolay ve verimli hale getiren bir Python kütüphanesidir. Web kazımada amaç projelerin daha düzenli ve ölçeklenebilir bir şekilde yönetilebilmesini sağlamaktadır.

Bilişim teknolojilerinin yaygınlaşması ile birçok sektörde dijital dönüşüm hızlanmıştır. Yeni web uygulamaları, e-ticaret platformları, sosyal medya, çevrimiçi eğitim ve telemedicine gibi alanlarda internetin etkisi ve önemi giderek artmaktadır (Kılınç vd., 2022). İnternet aracılığıyla nesnelerin birbirlerine bağlanarak insanların günlük hayatında giderek artan bir şekilde yer alması da

[1] Dr., Fırat Kalkınma Ajansı, abdullahesidir@yahoo.com, ORCID iD: 0000-0002-8106-1758,

günümüzde rastlanan bir değişimdir (Cevher, 2023). Öte yandan teknoloji ve bilgi; modern toplumların ekonomik büyüme, inovasyon ve rekabet gücü için kritik öneme sahip unsurlardır (Yoğunlu, 2022). Ar-Ge ve inovasyon, ulusal kalkınma ve teknolojik öncülük için kritik öneme sahip alanlardır. Bu kavramlar; teknolojik ilerleme, yeni ürünler/hizmetler, süreç iyileştirmesi ve rekabetçilik artışı gibi hedefleri amaçlamaktadır (Çubuk, 2023). İnternet, her geçen gün yeni ve gelişmiş web uygulamalarının geliştirilmesine olanak tanımaktadır. Bu uygulamalar, iş dünyası, eğitim, sağlık ve daha birçok alanda daha fazla işlevselliği ve erişilebilirliği artırabilmektedir. İnovasyon ve teknolojik ilerlemeler, işletmelerin rekabetçiliğini artırmaktadır. Daha iyi ürünler ve hizmetler sunma yeteneği, işletmelerin pazarda öne çıkmasına yardımcı olmaktadır.

Kaynakların etkin kullanımı, kalkınma açısından önemli bir faktördür (Eşidir ve Çubuk, 2023). Kaynakların etkin kullanılması, veri madenciliği projelerinin başarısı için kritik öneme sahiptir. Scrapy türünden rekabetçi teknolojilerin kullanımı, kaynakların etkin kullanılmasına yardımcı olmaktadır. Scrapy benzeri web kazıma araçları, veri madenciliği projelerinde veri toplama süreçlerini otomatize etmek açısından güçlü bir araçtır. Manuel, tek tek el ile veri toplama işlemleri, çok zaman alıcıdır ve oldukça da maliyetlidir. Web scraping araçları, bu süreçleri hızlandırmakta ve kaynakların daha etkin ve verimli kullanılmasına katkı sağlamaktadır.

Son zamanlarda veri bilimi alanında gelişen yeni yaklaşımlar ve teknolojiler, verinin bilgiye dönüştürülmesi sürecini hızlandırmakta ve daha kesin sonuçlar elde etme açısından büyük önem taşımaktadır (Kılınç ve Teke, 2020). Veri bilimi alanındaki yeni yaklaşımlar ve teknolojiler, veri analizinin hızını artırırken aynı zamanda daha kesin sonuçların elde etmektedir.

Scrapy, özellikle veri madenciliği, veri analizi ve otomasyon projelerinde başarı ile kullanılmaktadır. İnternet üzerinden büyük miktarlarda veri çekmek ve elde edilen veriyi kullanmak isteyenler için Scrapy güçlü bir araçtır. Scrapy kullanırken web sitelerinin kullanım şartlarına ve etik kurallara uyulması önemlidir, aksi takdirde yasal sorunlar ortaya çıkabilmektedir. Web kazıma işlemleri sırasında web sitelerine yük bindirilmesini önlemek için iyi bir izleme ve sınırlama mekanizması oluşturmak da gerekmektedir.

Web siteleri, botların veri toplamasını kısıtlayabilir veya yasaklayabilir. Bu nedenle, web kazıma yaparken bir web sitesinin "robots.txt" dosyasını kontrol etmek lazımdır. "robots.txt" dosyası, web sitelerinin botların hangi sayfalara erişebileceğini ve hangi sayfalara erişemeyeceğini belirleyen bir standart bir

txt dosyasıdır (Thomas ve Mathur, 2019). Scrapy, web sitelerinin robots.txt dosyalarını otomatik olarak kontrol etmektedir ve kullanıcıların izin verilen sayfalara erişimine olanak tanımaktadır. Eğer robots.txt dosyası belirli bir sayfanın taranmasına izin vermiyorsa, Scrapy o sayfayı atlamaktadır.

Çalışmanın temel amacı, web kazımanın neden önemli olduğunu ve bu alandaki önemli bir araç olan Scrapy'ın nasıl kullanılabileceğini anlatmaktır. Web kazıma, internetteki verileri toplamanın ve analiz etmenin önemli bir yoludur ve bu çalışma, bu sürecin neden kritik bir öneme sahip olduğunu açıklamaktadır.

Scrapy, web kazıma işlemlerini otomatize eden güçlü bir araçtır ve veri odaklı kararlar almak isteyenler ve internet üzerindeki içeriği analiz etmek isteyenler için önemlidir. Çalışma, web kazımanın önemini ve Scrapy'ın nasıl kullanılabileceğini anlatmakta ve veri madenciliği projelerinin nasıl geliştirilebileceği konusunda bir örnek sunmaktadır.

2. PYTHON PROGRAMLAMA DİLİ VE WEB ÖRÜMCEKLERİ (BOTLAR)

Python, birçok alanda etkin olarak kullanılan bir programlama dilidir (Malkoç, 2012). Python, web uygulamalarından bilimsel uygulamalara kadar çok geniş bir yelpazede yazılım geliştirme amacı ile kullanılmaktadır. Python, yüksek seviyeli ve nesne yönelimlidir ve Python ile birçok platformda uygulama geliştirmek mümkündür. Python, öğrenmesi kolay ve aynı zamanda güçlü bir yazılım dilidir. Veri yapılarına ve nesne yönelimli programlamaya basit ve etkili bir biçimde yaklaşmaktadır. Zarif söz dizisi ve dinamik yapısı sayesinde Python; birçok platformda komut dosyaları oluşturmak ve hızlı uygulama geliştirmek açısından ideal bir dildir (python.org, 2023).

Python, birçok programlama dilinden farklı olarak derlenmeye ihtiyaç duymadan, yorumlanabilir şekilde çalışmaktadır. Böylece, Python programları daha hızlı geliştirilip, çalıştırılabilmektedir. Fakat Python kodları yorumlanırken, Python yorumlayıcısının bilgisayarda yüklü olması gerekmektedir (Turan, 2023). Yorumlanabilir bir programlama dili, kaynak kodunun doğrudan çalıştırılabilmesini sağlayan bir yorumlayıcı tarafından işlenmektedir. Kaynak kodunun önce bir derleme aşamasından geçirilmesine gerek duyulmamaktadır. Yorumlayıcı, kodu satır satır okur ve anında çalıştırır. Python'ın yorumlanabilir bir dil olması, kod geliştirme sürecini hızlandırmakta ve hata ayıklamayı kolaylaştırmaktadır.

Python dilinde daha az satırla ve daha okunaklı kod yazılabilmektedir. Python kod yazma sürecini daha hızlı ve daha verimli hale getiren bir yapı sunmaktadır (Tratt, 2009). Python, web geliştirme, veri analizi, yapay zekâ, bilimsel hesaplama, otomasyon ve benzeri birçok alanda başarılı bir şekilde kullanılmaktadır. Python sayesinde, tekrarlayan ve zaman alan veri toplama işlemleri otomatik olarak yapılabilmektedir. Bu avantajlı yapı, büyük miktarlardaki veri toplama işlemleri veya düzenli şekilde güncellenen verilerin izlenmesi gereken durumlar için oldukça değerlidir.

Python programlama dili, web örümcekleri veya botlar gibi otomasyon yazılımlarının geliştirilmesi için oldukça yaygın ve etkili bir dil olarak kullanılmaktadır. Web örümcekleri veya botlar; web sitelerini otomatik olarak taramak, veri çekmek veya belirli görevleri yerine getirmek amacıyla kullanılmaktadır. Python; web örümcekleri ve veri toplama işlemleri için güçlü bir programlama dilidir ve bünyesinde bu amaç için birçok kütüphane barındırmaktadır. Python botlarının hazırlanması oldukça verimli ve etkili bir şekilde gerçekleştirilebilmektedir (Turan, 2023). Özellikle Requests, BeautifulSoup ve Scrapy benzeri Python kütüphaneleri web tarama ve veri çekme işlemlerini büyük ölçüde kolaylaştırmaktadır. Botların etik kurallara ve yasal düzenlemelere uygun şekilde kullanımı önemlidir. Python, istikrar konusunda kendisini kanıtlamış olgun bir programlama dilidir (python.org, 2023). Python, sürekli olarak güncellenmekte ve geliştirilmektedir. Botların istikrarlı çalışabilmesi için Python'ın güncel sürümlerinin kullanılması gerekmektedir. Python web botları ve veri toplama konusunda avantajlı bir dildir.

Web örümcekleri (web crawlers) ve botlar, internet üzerindeki içerikleri otomatik olarak tarayan ve toplayan yazılımlardır. Web örümcekleri, belirli bir başlangıç URL'sinden başlayarak internet üzerindeki diğer sayfaları ve bağlantıları taramak için tasarlanmıştır. Web örümcekleri, belirlenen sayfaları ziyaret eder ve içerikleri bilgisayara indirir. İçerik türleri metinler, resimler, videolar, ses dosyaları veya diğer medya türleri olabilir. Web örümcekleri, indirilen verileri geçici veya kalıcı şekilde saklayabilirler. Elde edilen veriler daha sonra kullanıcıların arama sonuçlarına daha hızlı bir biçimde erişmelerini sağlamak amacıyla kullanılmaktadır (Özdemir ve Gürcan, 2020).

Botlar, web siteleri veya uygulamalarda kullanıcılarla etkileşimde bulunabilirler. Sohbet botları, kullanıcıların sorularını yanıtlayabilir, talimatları takip edebilir ve çeşitli görevleri yerine getirebilir. Sosyal medya botları, otomatik olarak içerik paylaşabilir, beğenileri ve yorumları yönetebilir, takipçi sayılarını artırabilir ve sosyal medya hesaplarını yönetebilir. Veri toplama botları, internet

üzerindeki belirli bilgileri veya veri kümelerini otomatik olarak çekebilirler. Pazar araştırmaları, fiyat izleme veya haber toplama gibi çeşitli amaçlar için kullanılabilirler. Web örümcekleri ve botlar, internetin büyüklüğü ve karmaşıklığı göz önüne alındığında çok önemli araçlar olduğu ortaya çıkmaktadır. Bu yazılımlar; veri toplama, içerik indeksleme, otomasyon ve kullanıcı etkileşimi gibi birçok alanda işleri kolaylaştırmakta ve verimliliği artırmaktadır.

3. SCRAPY KÜTÜPHANESİ VE VERİ MADENCİLİĞİ

Scrapy, web tarama ve veri çekme işlemlerini otomatize etmek için kullanılan açık kaynaklı bir Python kütüphanesidir. Scrapy'nin temel amacı; kullanıcıların belirledikleri web sitelerini otomatik olarak gezmesine, veri çekmesine ve bu veriyi işlemesine olanak tanımaktır. Scrapy; veri madenciliği, veri analizi, otomasyon, web tarayıcı testleri ve daha birçok alanda oldukça kullanışlı bir araçtır (Scrapy 2.11 Documentation, 2023). Scrapy'nin temel özellikleri aşağıdaki gibi sıralanabilir:

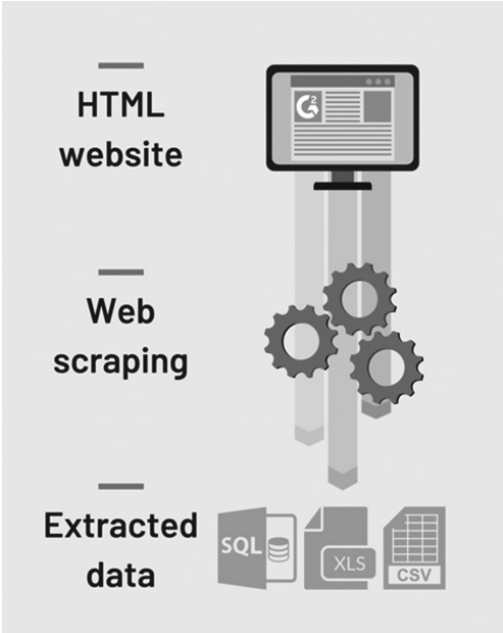
İleri Düzey Tarayıcı Yetenekleri: Scrapy, HTTP isteklerini yönetmek, web sitelerini gezinmek ve sayfalar arasında gezinirken veri çekmek için gelişmiş yeteneklere sahiptir. Bu sayede karmaşık web sitelerini rahatlıkla taramak mümkündür.

Esnek Veri Çekme: Scrapy, web sitelerinin yapısını analiz etmek için XPath veya CSS seçicilerini kullanarak veri çekme işlemlerini kolaylaştırmaktadır. İstenen belirli veriler hedeflenip çekilebilmektedir.

Otomatik Hata Yönetimi: Web sitelerine erişim sırasında oluşabilecek hatalar ele alınıp otomatik olarak yeniden denenebilme benzeri hata yönetimi stratejileri sunulmaktadır (Yıldız, 2019).

Paralel İşleme: Scrapy, aynı anda birden fazla sayfayı taramak için çoklu iş parçacığı veya çoklu süreç kullanarak hızlı veri çekme işlemleri gerçekleştirebilmektedir.

Veri Temizleme ve Dönüştürme: Çekilen verileri düzenlemek, temizlemek ve dönüştürmek için kullanıcı tarafından tanımlanan işlevlerin kullanılmasına izin verilmektedir. Scrapy, veri çekme işlemlerini hızlı ve verimli bir şekilde yapmaktadır. İnternet siteleri ile etkileşim kurulabilmekte ve form gönderilebilmektedir. Scrapy projeleri, spider adı verilen bileşenler ile birlikte çalışmaktadır. Spiderlar; hangi web sitelerinin ziyaret edileceğini, hangi verilerin çekileceğini ve nasıl işleneceğini tanımlamaktadır.



Şekil 1. Web Scrapy (Kaynak: Arzu Yıldız, 2019).

Scrapy, çeşitli veri madenciliği projeleri için kullanılan oldukça güçlü ve esnek bir araçtır (Yıldız, 2019). Örnek olabilecek bazı Scrapy projeleri aşağıda gösterilmiştir:

Pazar Araştırması: Scrapy, rekabetçi pazar analizlerinde kullanılabilir. Rakip firmaların ürünleri, fiyatları, kampanyaları ve müşteri geri bildirimleri gibi verileri toplamak amacıyla Scrapy kullanılabilir. Elde edilen bu ve benzeri veriler, bir işletmenin rekabet avantajını anlamasına ve stratejilerini optimize etmesine yardımcı olabilir.

Ürün ve Hizmet Geliştirme: Scrapy ile toplanan veriler, yeni ürünler veya hizmetler geliştirmek amacı ile kullanılabilir. Müşteri ihtiyaçlarını anlamak ve pazardaki eksiklikleri belirlemek için veri madenciliği yapılabilir.

Fiyatlandırma Stratejileri: Rakip firmaların fiyatları ve promosyonları hakkında veri toplamak, doğru fiyatlandırma stratejileri oluşturmak için yardımcı olabilir.

Müşteri Analizi: Scrapy ile sosyal medya ve çevrimiçi platformlardan müşteri geri bildirimleri ve incelemeleri toplanabilir. Bu veriler, müşteri davranışlarını ve tercihlerini anlamak amacıyla kullanılabilir.

Rekabet İzleme: Rekabetçi teknolojiler, rakip firmaların web sitelerini düzenli olarak izleyebilir ve değişiklikleri takip edebilir. Rakip stratejilerini anlamak ve hızla yanıt vermek önemlidir.

Veri Madenciliği Algoritmaları: Scrapy ile toplanan veriler, veri madenciliği algoritmalarıyla işlenerek daha fazla içgörü elde edilebilir. Özellikle büyük veri setlerini analiz etmek ve desenler bulmak için kullanışlıdır.

Sosyal Bilimler Araştırmaları: Scrapy ile sosyal medya platformlarından veya forumlardan toplanan veriler, sosyal bilimlerdeki araştırmalar için kullanılabilir, örneğin toplumsal eğilimleri veya kamusal görüşleri incelemek için.

Eğitim ve Araştırma Verileri: Eğitim kaynaklarından (üniversite web siteleri, öğrenci değerlendirmeleri) veya akademik makalelerden veri çekmek, araştırmacılar ve eğitimciler için önemlidir.

Hava Durumu ve Coğrafi Veriler: Hava durumu verileri, coğrafi konum verileri ve haritalar, hava tahminleri, jeolojik bilgiler ve konum tabanlı hizmetler için kullanılabilir.

Sağlık ve Tıbbi Veriler: Sağlık kaynaklarından tıbbi bilgiler, hastane verileri, ilaç fiyatları ve sağlık trendleri hakkında veri çekmek, sağlık alanında araştırma ve analiz yapmak için kullanılır.

Scrapy, yukarıdaki ve benzeri birçok veri madenciliği uygulamalarında başarı ile kullanılabilmektedir (Scrapy 2.11 Documentation, 2023).

Scrapy kütüphanesinin en güçlü özelliklerinden biri asenkron işlem yapabilme yeteneğidir. Asenkron programlama, aynı anda birden fazla işlemi eşzamanlı olarak yürütebilme yeteneğini ifade eder ve bu, web tarama ve veri çekme işlemlerinde büyük bir avantaj sağlar. Scrapy'nin asenkron özelliği, özellikle büyük ve karmaşık web sitelerinden veri çekme işlemlerinde çok işe yaramaktadır. Bu özellik, veri madenciliği, web tarama, SEO analizi ve benzeri uygulamalar için Scrapy'ın güçlü bir tercih olmasını sağlamaktadır. Scrapy'ın kapsamlı bir dökümantasyonu vardır ve geniş bir kullanıcı topluluğuna sahiptir.

Bazı web siteleri, robots.txt dosyasını kullanarak hangi sayfaların ve verilerin kazınabileceğini sınırlamaktadır. Bu dosyanın önerilen kurallarına uymak önemlidir. Scrapy, ROBOTSTXT_OBEY ayarı ile bu kuralları otomatik olarak izleyebilir (Sapaz ve Tarcan, 2018).

ROBOTSTXT_OBEY = True ayarı yapılmalıdır.

Büyük web sitelerini taramak zaman alabilmektedir ve böyle durumlarda web sunucusu tarafından zaman aşımı hatası alınabilmektedir. Bu durumda, Scrapy

ayarlarını güncellemek ve talepleri daha yavaş hızlarda yapmak veya sayfalar arası geçişlerde daha fazla bekleme eklemek gibi önlemler almak gerekir. Bazı web siteleri, botları ve web örümceklerini tespit etmek ve engellemek için anti-kazıma önlemleri alabilirler. Bu durumda, IP adresini veya kullanıcı ajanını değiştirmek gerekebilir. Ayrıca, otomatik Captcha çözme bu türden sıkıntıları çözebilir.

Web sitesi sahipleri, yoğun veya zararlı web kazıma aktivitelerini engellemek için IP tabanlı banlar uygulayabilirler. Bu nedenle, aynı IP adresinden çok sayıda talep gönderilirken daha dikkatli olunmalıdır. IP rotasyonu veya proxy kullanımı bu tür sorunları aşmada yardımcı olabilmektedir.

Web siteleri, kullanıcı ajanı başlığına veya çerezlere bağlı olarak web örümceğini tanıyabilir ve engelleyebilir. Kullanıcı ajanı başlığı ve gerektiğinde çerezler ayarlanarak bu tür sınırlamalar aşılabilmektedir. Scrapy ile web kazıma uygulamaları geliştirirken engelleri aşmak için dikkatli planlama, esneklik ve hata işleme stratejileri geliştirmek gereklidir. Bunlara dikkat edilerek Scrapy ile başarılı web kazıma uygulamaları oluşturulabilmektedir.

Scrapy'nin desteklediği veri saklama formatları şunlardır:

JSON: JSON (JavaScript Object Notation), verilerin hafif ve okunabilir bir şekilde saklanmasını sağlamaktadır. JSON formatı, verilerin daha sonra kolayca okunabilmesi için uygun bir seçenektir.

CSV: Kazılan veriler CSV (Comma-Separated Values) formatında saklanabilmektedir. CSV, verileri tablo formatında saklamak için kullanılmaktadır ve birçok veri analizi ve işleme aracı tarafından desteklenmektedir.

XML: Scrapy, verileri XML (eXtensible Markup Language) formatında saklayabilmektedir. XML, belirli bir yapıda verileri saklamak için kullanılmakta ve veriler doğrusal bir şekilde temsil edilmektedir.

Veritabanları: Scrapy, kazılan verileri SQLite, MySQL, PostgreSQL gibi veritabanlarına kaydetmeye olanak tanır. Böylece, veriler daha karmaşık bir yapıda ve uzun vadeli bir şekilde saklanabilmektedir.

JSON Lines: JSON Lines formatı, JSON benzeri bir yapıya sahiptir, ancak her satırda bir JSON nesnesi bulunur. Bu format, büyük veri kümesi işlemleri için uygundur.

Scrapy, bu formatların çoğunu kullanarak verileri saklar ve ilgili veri kazıma projeleri yürütülebilir. Verilerin belirlenen formatta saklanması için, Scrapy ayarlarının ve kodların uygun şekilde yapılandırılma gereklidir. Scrapy, veri

çekme ve saklama süreçlerini kolaylaştırma açısından birçok özellik ve esneklik sunmaktadır (Zyte, 2023).

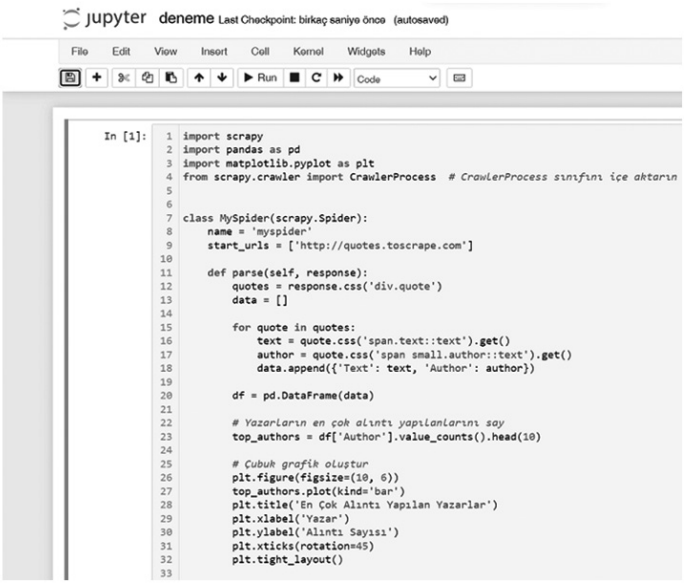
4. SCRAPY İLE ÖRNEK WEB KAZIMA UYGULAMASI

Web sitelerinde bulunan açık veriler (open data), kısıtlayıcı kullanım koşullarına tabi olmaksızın herkes tarafından ücretsiz olarak kullanılabilirler (Karasakal vd., 2022). Açık veriler, çeşitli alanlarda kullanılır, özellikle veri analizi, veri görselleştirme, araştırma, kamu politikalarının oluşturulması, işletme stratejilerinin geliştirilmesi ve daha birçok alanda. Açık verilerin serbestçe erişilebilir ve kullanılabilir olması, toplumun bilgiye daha geniş bir erişimine ve inovasyona olanak tanır. Scrapy, büyük veri çekme projeleri ve web sitesi taramaları için güçlü bir araçtır ve veri madenciliği, araştırma, rekabet analizi ve daha birçok uygulama alanında kullanılabilir (Scrapy 2.11 Documentation, 2023).

Python'da import komutu, dışarıdan bir modül veya kütüphane içeri aktarılarak kullanılmasını sağlayan temel bir yapıdır (Tek, 2023). Bu modüller, Python'ın standart kütüphanesinin bir parçası olmadığı için, kullanmadan önce modüllerin bilgisayara indirilmesi gerekmektedir. İndirme işlemi Windows işletim sistemi kullanıcıları için komut istemcisine (CMD) girilen komutlar aracılığıyla gerçekleştirilmektedir. Scrapy kütüphanesini indirmek için Windows'ta CMD penceresi açılıp şu komut girilir:

pip install scrapy

Bu komut, Python'un paket yöneticisi olan pip kullanılarak Scrapy'ı indirecek ve kuracaktır. İndirme işlemi tamamlandıktan sonra, internet üzerinden veri çekme işlemine geçilebilir (Tek, 2023). Şekil 2'de Jupyter Notebook çalışma ortamı ve web kazıma örneği için yazılmış Python kodları gösterilmektedir.



Şekil 2. Jupyter Notebook Çalışma Ortamı ve Web Kazıma Örneği İçin Yazılan Python Kodları

Formun ÜstüScrapy kullanılarak bahsedilen işlemleri gerçekleştiren bir Python projesi gerçekleştirilecektir. 'http://quotes.toscrape.com', web kazıma (web scraping) örneklerinde sıkça kullanılan kurgusal bir web sitesidir. Bu web sitesini genelde web kazıma pratiği yapmak isteyenler kullanmaktadır. Web sitesi farklı sayfalar içermektedir ve her sayfa farklı yazarlara ait alıntıları içermektedir. Web sitesinin HTML yapısı basit ve okunaklıdır. Web kazıma işlemleri yapılırken HTML kodlarını analiz etmek ve verileri çekmek için iyi bir öğrenme kaynağıdır. Öncelikle, import işlemi ile gerekli kütüphanelerin içe aktarılması gerekmektedir.

import scrapy

import pandas as pd

import matplotlib.pyplot as plt

from scrapy.crawler import CrawlerProcess

CrawlerProcess sınıfı içe aktarılıyor

Bu kütüphaneler, web scraping işlemleri için gerekli araçları ve veri manipülasyonu ile görselleştirme işlemleri için temel araçları sağlamaktadır.

scrapy: Bu kütüphane, web sitesinden veri çekmek için kullanılan güçlü bir web kazıma (web scraping) kütüphanesidir. Scrapy, web sitelerini ziyaret etmek, verileri çekmek ve işlemek için birçok araç ve yöntem sunmaktadır.

pandas: Pandas, Python'da veri analizi ve işlemek için yaygın olarak kullanılan bir kütüphanedir. Veri çerçeveleri (dataframes) oluşturulmasına, verilerin filtrelenmesine, dönüştürülmesine ve analiz edilmesine olanak sağlamaktadır.

matplotlib: Verileri görselleştirmek için kullanılmaktadır. Çeşitli grafikler, grafikler ve görsel öğeler oluşturulur.

CrawlerProcess: Scrapy'nin bir parçası olan bu sınıf, Scrapy örümceklerini (spiders) başlatmak ve çalıştırmak için kullanılmaktadır.

```
class MySpider(scrapy.Spider):
    name = 'myspider'
    start_urls = ['http://quotes.toscrape.com']
```

class MySpider(scrapy.Spider): Bu satır, MySpider adında yeni bir örümcek sınıfı tanımlamaktadır. Bu sınıf, Scrapy tarafından sağlanan scrapy.Spider sınıfını miras alır ve bu sınıfı genişletir.

name = 'myspider': Örümceğin adı 'myspider' olarak ayarlanır. Her örümceğin benzersiz bir adı olmalıdır. Böylelikle Scrapy bu örümceği diğer örümceklerden ayırabilir.

start_urls = ['http://quotes.toscrape.com']: Örümceğin başlaması gereken URL'yi belirtir. Örümcek, bu URL'den veri çekmeye başlayacaktır.

```
def parse(self, response):
quotes = response.css('div.quote')
data = []
```

Yukarıdaki Python kodu, Scrapy örümceği sınıfı içinde tanımlanan parse adlı bir yöntemi içermektedir. parse yöntemi, örümceğin bir web sayfasını ziyaret ettiğinde çalışan ana işlevi içermektedir.

def parse(self, response): Bu satır, parse adında bir yöntem tanımlamaktadır ve bu yöntemin self (örümcek nesnesi) ve response (web sayfası yanıtı) parametrelerini kabul ettiğini belirtir.

quotes = response.css('div.quote'): Bu satır, response içindeki HTML belgesindeki tüm öğelerini seçer ve bu öğeleri quotes değişkenine atar. Bu, web sayfasındaki alıntıları temsil eden HTML öğelerini seçmek için kullanılan bir CSS seçici (CSS selector) kullanımıdır.

data = []: Bu satır, boş bir liste olan data adında bir değişken oluşturmaktadır. Bu liste, alıntılanan verileri saklamak için kullanılmaktadır.Formun Üstü

```
for quote in quotes:
        text=quote.css('span.text::text').get()
author = quote.css('span small.author::text').get()
data.append({'Text': text, 'Author': author})
df = pd.DataFrame(data)
```

Bu Python kodu, web sayfasındaki alıntıları çekerek ve bunları bir Pandas veri çerçevesine dönüştürerek verileri düzenlemek için kullanılır.

for quote in quotes: Bu döngü, quotes değişkenindeki her bir alıntı öğesini işlemek için kullanılır. Her bir alıntı öğesi, web sayfasındaki alıntıların bir temsilcisidir.

text = quote.css('span.text::text').get(): Bu satır, her bir alıntının metin içeriğini seçer ve text adlı bir değişkene atar. quote.css('span.text::text') ifadesi, alıntı öğesinin içindeki metin içeriğini temsil eden CSS seçicisini kullanır. .get() yöntemi, bu metin içeriğini alır.

author = quote.css('span small.author::text').get(): Bu satır, her bir alıntının yazarını seçer ve author adlı bir değişkene atar. Benzer şekilde, quote.css('span small.author::text') ifadesi yazar adını temsil eden CSS seçicisini kullanır ve .get() yöntemiyle bu adı alır.

data.append({'Text': text, 'Author': author}): Bu satır, her bir alıntının metin ve yazar bilgisini içeren bir sözlük oluşturur ve bu sözlüğü data adlı bir liste içine ekler. Her bir alıntı, bu liste içinde bir sözlük olarak temsil edilir.

df = pd.DataFrame(data): Bu satır, data adlı listenin üzerine bir Pandas veri çerçevesi (DataFrame) oluşturur. Veri çerçevesi, alıntı metinleri ve yazarları gibi verileri düzenlemek ve analiz etmek için kullanılabilir.

Sonuçta, bu kod parçası, web sayfasından çekilen alıntıları bir veri çerçevesine dönüştürür ve bu veri çerçevesini df adlı bir değişkende saklar.

```
# Yazarların en çok alıntı yapılanlarını say
top_authors = df['Author'].value_counts().head(10)
# Çubuk grafik oluştur
        plt.figure(figsize=(10, 6))
top_authors.plot(kind='bar')
plt.title('En Çok Alıntı Yapılan Yazarlar')
plt.xlabel('Yazar')
plt.ylabel('Alıntı Sayısı')
```

```
plt.xticks(rotation=45)
plt.tight_layout()
        # Grafiği göster
plt.show()
        process = CrawlerProcess()
        process.crawl(MySpider)
        process.start()
```

top_authors = df['Author'].value_counts().head(10): Bu satır, veri çerçevesinde bulunan yazarların en çok alıntı yapılanlarını sayar ve bu bilgileri top_authors adlı bir seri olarak saklar.

value_counts() yöntemi, her yazarın kaç kez alıntı yapıldığını sayar ve büyükten küçüğe sıralar. head(10) ise en çok alıntı yapılan ilk 10 yazarı alır.

plt.figure(figsize=(10, 6)): Bu satır, çubuk grafiği oluşturmak için bir Matplotlib figürü oluşturur ve grafiğin boyutunu (genişlik ve yükseklik) belirtir.

top_authors.plot(kind='bar'): Bu satır, top_authors Serisi üzerinde çubuk grafik çizmek için kullanılır. Bu, yazarların en çok alıntı yapılanlarını gösterir.

plt.title('En Çok Alıntı Yapılan Yazarlar'): Bu satır, grafiğin başlığını ayarlar.

plt.xlabel('Yazar'): Bu satır, x ekseni (yatay ekseni) etiketler.

plt.ylabel('Alıntı Sayısı'): Bu satır, y ekseni (dikey ekseni) etiketler.

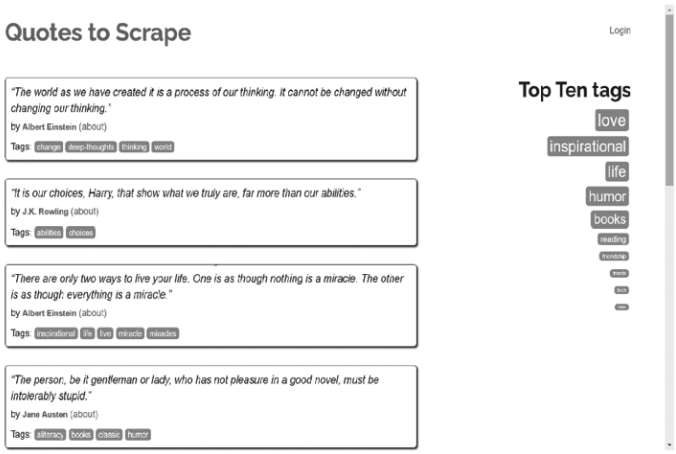
plt.xticks(rotation=45): Bu satır, x ekseni etiketlerinin 45 derece dönerek daha okunabilir olmasını sağlar.

plt.tight_layout(): Bu satır, grafik öğelerini düzenlemek ve daha iyi bir görünüm sağlamak için kullanılır.

plt.show(): Bu satır, grafiği görüntüler. Grafiği görüntülemek için bu kod parçasının sonuna gelindiğinde çağrılır.

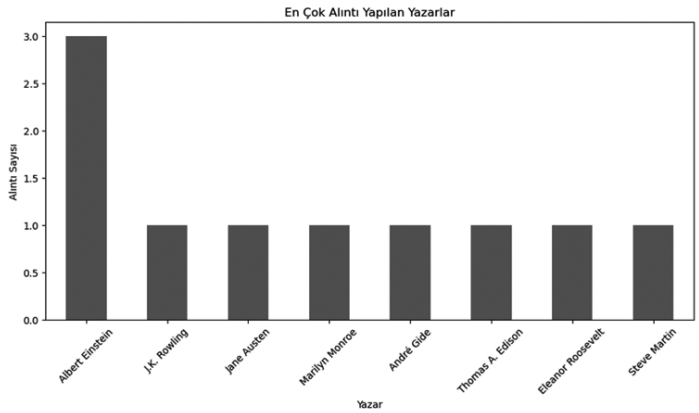
Son olarak, CrawlerProcess() ile bir process oluşturulur ve process. crawl(MySpider) ile örümcek başlatılır. Daha sonra process.start() ile örümcek çalıştırılır. Bu kod, örümceğin web sitesini ziyaret etmesini ve verileri çekmesini başlatır. Veriler çekildikten sonra, bu kod ile çubuk grafik oluşturulur ve gösterilir.

Şekil 3'te quotes.toscrape.com web sitesi gösterilmiştir. Bu site, başlıca ünlü yazarlardan alınmış alıntıları içerir ve kullanıcılar bu alıntıları çeşitli kategorilere göre listeleyebilirler. Bu web sitesi, Python programlamadaki veri çekme örneklerini uygulamak için yaygın olarak kullanılmakta olan bir web sitesidir.



Şekil 3. "quotes.toscrape.com" Web Sitesi

Şekil 4'te en çok alıntı yapılan yazarlar grafik ortamında gösterilmiştir. Kod, alıntıları çekerken her yazarın kaç alıntı yaptığını sayar ve bu veriyi kullanarak en çok alıntı yapılan ilk 10 yazarı bir çubuk grafikte göstermektedir. Çubuk grafik, Jupyter Notebook içerisinde görüntülenmektedir.



Şekil 4. En Çok Alıntı Yapılan Yazarlar

5. SONUÇ VE DEĞERLENDİRME

Çalışmanın amacı ve içeriği, Scrapy'ın temel işleyişini ve web kazıma projelerindeki etkinliğini tanıtmaya yöneliktir. Çalışma, Scrapy ve Python›un veri analizi kütüphanelerini kullanarak web kazıma ve veri analizi işlemlerinin nasıl

gerçekleştirileceği konusunu ayrıntılı olarak işlemektedir. Bu türden işlemlerin uygulanması için de iyi bir örnek sunulmuştur. Örnek Scrapy projesi, ‹http://quotes.toscrape.com› web sitesi üzerinde geliştirilmiştir. Bu web sitesi, özellikle Python programlama dili ile web kazıma pratiği yapmak isteyenler için ideal bir seçenektir. Çalışmada, açık verilerin önemini vurgulanmış ve Scrapy›ı indirme ve kurma işlemi gibi temel adımlar da açıklanmıştır.

Scrapy, web sitelerinden veri çekme sürecini otomatize etme açısından güçlü ve esnek bir araçtır. İleri düzey tarayıcı yetenekleri, esnek veri çekme özellikleri ve otomatik hata yönetimi gibi özellikleri; veri madenciliği, içerik kazıma ve rekabet analizi benzeri birçok alanda oldukça kullanışlıdır. Scrapy'ın açık kaynaklı olması ve geniş bir kullanıcı kesimini içermesi, sürekli olarak geliştirilmesine ve yeni özelliklerin eklenmesine olanak tanımaktadır. Ayrıca, programcılar Scrapy'ı kendilerine has olarak özelleştirebilir ve ihtiyaçlarına göre projelerine uyarlayabilirler.

Teknoloji, yeni ürünler ve teknolojik araştırmalarla ilgili arge çalışmaları, özel yatırımların miktarı ve etkinliği açısından gereklidir (Çubuk, 2021). Bu açıdan değerlendirildiğinde, Scrapy ve benzeri teknolojiler, işletmelerin rekabetçi kalmalarına ve yenilikçi ürünler geliştirilmesine yardımcı olmaktadır. Bu araçlar, işletmelerin rekabetçi kalmalarına ve teknolojik ilerlemeyi yakalamalarına katkı sağlamaktadır.

Scrapy'nin asenkron yapısı, özellikle büyük ve karmaşık web sitelerini taramak veya birden çok kaynaktan veri çekmek açısından idealdir. Scrapy, web tarama ve veri çekme projelerinde verimliliği artırmak amacı ile sıkça tercih edilmektedir. Scrapy'ın resmi dokümantasyonu, bu kütüphanenin temellerini öğrenmek ve karmaşık işlemleri gerçekleştirmek açısından güzel ve tavsiye edilen bir kaynaktır (https://docs.scrapy.org/en/latest/). Birçok çevrimiçi platformda, Scrapy hakkında dersler ve eğitimler sunulmaktadır. Udemy, Coursera ve edX benzeri platformlarda Scrapy eğitimleri verilmektedir. Scrapy ve veri analizi becerilerini geliştirmek için gerçek projeler üzerinde çalışma yapmak önemlidir. Kişiye özel projeler oluşturmak veya açık kaynak veri setlerini analiz ederek pratik yapmak, konu hakkında kişisel programlama becerilerini geliştirecektir.

Scrapy web kazıma aracı olarak, birçok projede oldukça etkili olabilmektedir. Ancak doğal olarak bazı sınırlamalara ve zorluklara sahiptir. Bu sınırlamalar aşağıda belirtildiği gibi olabilir (Scrapy 2.11 Documentation, 2023).

Site Yapısı Değişiklikleri: Web sitelerinin yapısı değişebilir ve böylece web örümceğinin çalışması etkilenebilir. Site değişikliklerine hızlı bir şekilde adapte olabilme yeteneği önemlidir.

IP Engelleme ve Hız Sınırlamaları: Web siteleri, çok hızlı talepleri veya aynı IP adresinden gelen talepleri engelleyebilmektedir. IP rotasyonu ve hız sınırlamalarının düzgün ayarlanması bu sorunları aşmakta yardımcı olabilmektedir.

Doğru Veri Seçimi: Web siteleri genelde çok fazla veri içermektedir ve mevcut veriler arasından istenilen verilerin doğru bir şekilde seçilmesi gereklidir. Doğru CSS veya XPATH seçicilerinin kullanılması gereklidir.

Veri Saklama ve Temizleme: Web kazıma işlemi sonrasında verilerin düzenlenmesi ve temizlenmesi gerekebilir. Böylece veri analizi ve raporlama süreçleri kolaylaşır.

Özellikle dinamik web sitelerini kazımak, sayfa içeriğinin yüklenmesi ve javascript ile etkileşim gerektiren işlemler ile başa çıkmak Scrapy açısından zor olabilmektedir. Bu tür zorluklar, geliştiricilerin alternatif kazıma yöntemleri veya daha özelleştirilmiş çözümler aramalarına neden olmaktadır.

Yapay zekâ ve makine öğrenmesi, web kazıma süreçlerini daha akıllı hale getireceği değerlendirilmektedir. Yakın gelecekte, derin öğrenme algoritmaları kullanımı ile metin verilerinin ötesine geçilerek görsel ve ses verilerinin de kazılması mümkün olacaktır. Bu alandaki gelişmeler, Scrapy'ın yeteneklerini arttıracaktır (Zyte, 2023). Scrapy, daha etkili web kazıma projeleri geliştirebilmek için sürekli şekilde geliştirilmeye ihtiyaç duymaktadır. Gelecek yıllarda web kazıma işlemleri; daha fazla otomatikleşme, yapay zekâ kullanımı ve daha gelişmiş veri analizleri ile şekilleneceği beklenmektedir. Scrapy, internetten veri çekme işlemlerini daha akıllı ve verimli hale getirmek için geliştirilmeye ihtiyaç duymaktadır. Scrapy ve benzeri web kazıma araçları, gelecekteki web kazıma işlemlerine uyum sağlamak ve veri çekme projelerini başarıyla yönetmek açısından önem arz etmektedir. Bu türden araçların sürekli gelişim gösteren internet dünyasında veriye erişimin temel bir bileşeni olarak kalacakları değerlendirilmektedir.

KAYNAKÇA

Alanoğlu, Z. & Akcayol, M. A. (2023). Web Tarayıcılarında Tohum URL Seçimi ve Performans Analizi: Kapsamlı Bir İnceleme. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, 11 (3), 1399-1423. DOI: 10.29130/dubited.1097123

Cevher, M. F. (2023). Bibliometric Analysis of the Internet of Things from a Marketing Perspective. *Journal of Research in Business,* 8(1), ss. 153-170.

Çubuk, M. (2021). *Çok Kriterli Karar Verme Yöntemleri ile İllerin Yatırım Ortamlarının Karşılaştırılması.* Ankara, Gazi Kitabevi, s. 56.

Çubuk, M. (2023). R&D and Innovation Map of Turkey: Hybrid Model Approach, *Turkish Journal of Science & Technology* Research Paper, 18(2), ss. 487-502.

Eşidir, K. A. & Çubuk, M. (2023). Çoklu Doğrusal Regresyon Analizi ile Bölgesel Kaynakların İhracat Fiyatlarının İncelenmesi: Yüksek Karbonlu Ferrokrom Örneği, *Bölgesel Kalkınma Dergisi*, 01 (01), ss. 104-116.

Karasakal, S., Doğan, O., Yücesoy, S., (2022). *Airbnb Yorumları Üzerine Bir Araştırma: Antalya Örneği,* 22. Ulusal Turizm Kongresi, 27-29 Ekim 2022 Burdur, ss. 417- 429.

Kılınç, M. & Aydın, C. (2019). Web Tabanlı İş Analitiğinin, İşletmelerdeki Kısa ve Orta Vadede Karar Verme Mekanizmasına Olan Etkisinin Araştırılması. *Yönetim Bilişim Sistemleri Dergisi*, 5 (1), 64-85.

Kılınç, M. & Teke, İ. (2020). Veri Görselleştirmenin Bilgi Sistemlerinde Kullanımı: Web Tabanlı Mezun Bilgi Sistemi Örneği. *Manisa Celal Bayar Üniversitesi Sosyal Bilimler Dergisi,* 18 (1), 96-109. DOI: 10.18026/cbayarsos.584804

Kılınç, M., Aydın, C. & Tarhan, Ç. (2022). Türkiye'de Sosyal ve Dijital Girişimcilik: Veri Kazıma Teknikleriyle Kitle Fonlaması Platformlarının İçerik Analizi. *Acta Infologica*, 6 (1), 83-97. DOI: 10.26650/acin.997640

Malkoç, B., (2012). *Temel Bilimler ve Mühendislik Eğitiminde Programlama Dili Olarak Python, Akademik Bilişim*.12- XIV. Akademik Bilişim Konferansı Bildirileri, 1- 3 Şubat 2012, Uşak Üniversitesi, ss. 201-210.

Özdemir, Ş., & Gürcan, H. İ., (2020). Derin Web ve Karanlık Web›te Sağlık Ticareti. 6. Sağlık İletişimi Sempozyumu, Eskişehir, Turkey.

Python 3.11.5 Belgelendirmesi, https://docs.python.org/tr/3/, Erişim Tarihi: 24.09.2023.

Sapaz, B., & Tarcan, G. Y. (2018). Arama Motoru Optimizasyonu (SEO): Özel Hastane Web Siteleri Üzerine Bir İnceleme, III. Uluslararası Stratejik Araştırmalar Kongresi, 03-05 Mayıs 2018, Çorum.

Scrapy 2.11 Documentation, (2023). https://docs.scrapy.org/en/latest/, Erişim Tarihi: 21.09.2023

Tek, M., (2023). Python ile Veri Çekme-BeautifulSoup-Selenium-Scrapy, Rexven Academy, https://www.udemy.com/course/python-ile-veri-cekme-beautifulsoup-requests-selenium-scrapy/, Erişim Tarihi: 25.09.2023.

Thomas, D. M. ve Mathur S., (2019). *Data Analysis by Web Scraping using Python, Proceedings of the Third International Conference on Electronics Communication and Aerospace Technology* [ICECA 2019], IEEE Conference Record # 45616; IEEE Xplore, ISBN: 978-1-7281-0167-5, ss. 450, 454.

Tratt, L., (2009). Dynamically Typed Languages, Advances in Computers, 77: ss. 149-184.

Turan, S., (2023). Python ile Sıfırdan İleri Seviye Programlama, https://www.udemy.com/course/python-dersleri/, Erişim Tarihi: 21.09.2023.

Turan, S., (2023). Python ile Sıfırdan İleri Seviye Python Programlama, Sıfırdan İleri Seviye Python Dersleri, Veritabanı,Veri Analizi,Bot Yazımı,Web Geliştirme(Django) , https://www.sadikturan.com/python-dersleri

Yıldız A, (2019). Scrapy ile Web Scraping(Web kazıma) Nasıl Yapılır?, https://medium.com/@arzuyildiz/scrapy-ile-web-scraping-web-kaz%C4%B1ma-nas%C4%B1l-yap%C4%B1l%C4%B1r-%EF%B8%8F-841cf7645c10, Erişim Tarihi: 21.09.2023

Yoğunlu, A., (2022). *Yenilik Ekosistem Yaklaşımına Dayalı Teknoloji Geliştirme Bölgeleri*, Gazi Kitabevi, ISBN: 978-625-8413-72-4, Ankara, ss. 177.

Zyte. (2023). Scrapy, New York. URL: https://www.zyte.com/, Erişim Tarihi: 27.09.2023.