YAZILIM GELİŞTİRME LABATUVARI-1

Erol Malkoç

Bilişim Sistemleri Mühendisliği

221307009
erolmalkoc04@gmail.com

Osman Musap Cuha

Bilişim sistemleri Mühendisliği
221307030

musabcuha@gmail.com

Özet

Bu projede, İstanbul'un hava durumu tahminlerini yapmak amacıyla yapay zeka kullanılarak bir model geliştirmeye çalışıyoruz. **Meteostat** veritabanından otomatik olarak veri çekilmesi için Selenium aracı kullanılmış ve veriler Python ile işlettik. Zaman serisi analizi yapılarak, sıcaklık, nem ve rüzgar gibi faktörlerin gelecekteki değerleri tahmin etmek için verileri çektik ve işledik. Bu çalışma, hava durumu tahminlerinin daha doğru ve verimli hale getirilmesi için gelecekteki projelere temel oluşturacaktır.

1. Giriş

Hava durumu tahmini, insan yaşamını ve çeşitli endüstriyel faaliyetleri doğrudan etkileyen önemli bir alandır. Bu projede, İstanbul için hava durumu tahminlerini yapay zeka (YZ) yöntemleriyle gerçekleştirmeyi amaçladık. Zaman serisi analizi ve makine öğrenimi teknikleri kullanılarak, geçmiş hava durumu verilerinden gelecekteki koşullar tahmin edilmeye çalışıyoruz. Veriler, **Meteostat** adlı açık hava durumu veritabanından çekilmiş ve işlenmiştir.[1-14]

2. Kullanılan Yazılım Dilleri

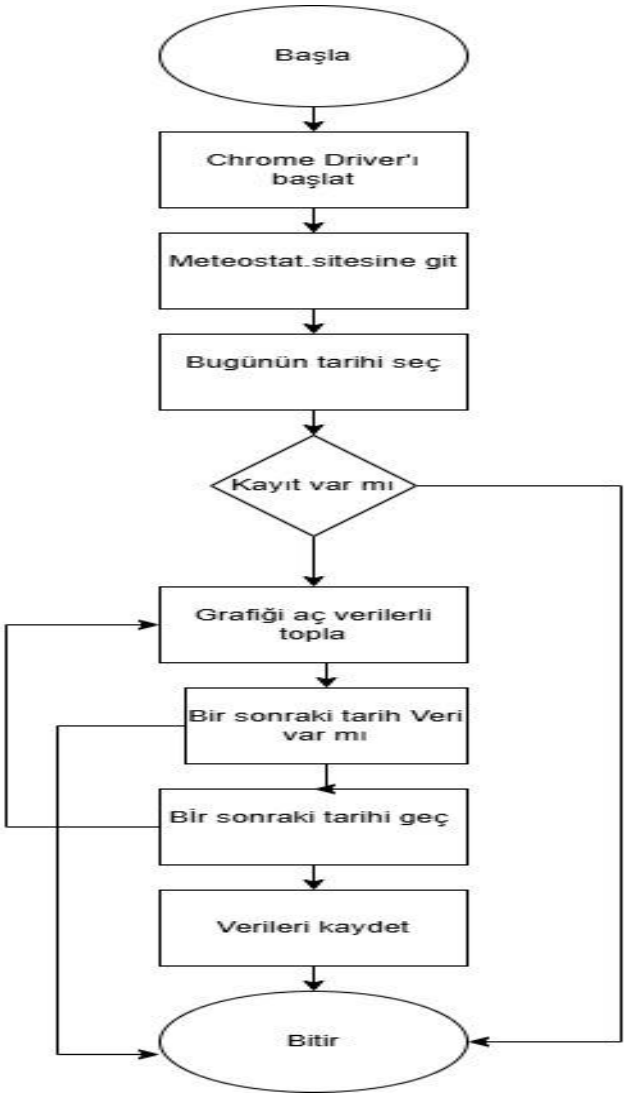
Projemizde veri toplama, işleme ve tahmin aşamalarında çeşitli yazılım araçları ve kütüphaneler kullanılmıştır:

- **Selenium**: Selenium, projenin önemli bir parçası olarak, web scraping işlemiyle verileri otomatik olarak elde etmede kullanılmıştır. Selenium, özellikle dinamik web sayfalarından veri çekmek için kullanılan popüler bir araçtır. Diğer statik scraping yöntemlerine kıyasla, tarayıcı otomasyonu ile sayfalar üzerindeki dinamik içerikleri yükleyip etkileşim kurabilme yeteneği sunar. Selenium, birden fazla programlama diliyle (Python, Java, C#, vb.) uyumlu çalışabilmesi sayesinde geniş bir kullanıcı kitlesine hitap eder. Projemizde Python dili ile birlikte kullanıldı. Python'un Selenium WebDriver arayüzü ile tarayıcılar üzerinde etkileşim kurmak, elementleri bulmak ve kullanıcı davranışlarını simüle etmek mümkün hale gelir. WebDriver, tarayıcıların açılması, butonlara tıklama, form doldurma ve sayfa kaydırma gibi işlemleri gerçekleştirebilen güçlü bir bileşendir.

  Projemizde, Meteostat'tan verilerin doğru ve düzenli bir şekilde çekilebilmesi için Selenium'un "find_element", "click" ve "send_keys" gibi komutları kullanılmıştır. Bu komutlar yardımıyla tarayıcı üzerinde çeşitli elementlerle etkileşime girilmiş, gerekli veriler toplanarak Excel formatına dönüştürülüp saklanmıştır.
  Selenium ayrıca "Explicit Wait" ve "Implicit Wait" gibi zamanlayıcılarla da desteklenmiştir. Bu bekleme mekanizmaları, dinamik içeriklerin tam olarak yüklenmesi ve sayfa yükleme sürelerinin farklılık göstermesi durumunda verilerin hatasız bir şekilde çekilmesine olanak tanır.[1,2,5,11]
- **Pthon**: Projede Python, veri işleme ve modelleme aşamalarında kilit rol oynamıştır. Python'un veri işleme kütüphaneleri olan Pandas ve NumPy kullanılarak veriler okunmuş, düzenlenmiş ve analizler için uygun hale getirilmiştir. Verilerin analiz ve görselleştirme süreçlerinde Excel formatına dönüştürülerek saklanması sağlanmıştır. Ayrıca, matplotlib ve seaborn gibi kütüphaneler yardımıyla verilerin görselleştirilmesi gerçekleştirilmiştir.[4-7]
- **Excel**: Verilerin işlenmesi ve görselleştirilmesi için Excel formatında saklanmıştır ..[12]

3. Veri Toplama Yöntemi

İstanbul'un hava durumu verileri, **Meteostat** web sitesinden **Selenium** kullanılarak otomatik olarak kazınmıştır. Bunun yanı sıra, **OpenWeather API** kullanılarak da ek hava durumu verileri çekilmiştir. Elde edilen tüm veriler, **JSON** formatında düzenlenmiş ve zaman serisi analizi için işlenmiştir. Bu süreçte, hem manuel web scraping hem de API üzerinden veri toplama yöntemleri bir arada kullanılmıştır. Çekilen veriler, **Excel** formatına dönüştürülerek analizlere uygun hale getirilmiştir.[5,10]



A. Web Scraping:

Projede, **Selenium** aracı kullanılarak Meteostat web sitesinden hava durumu verileri otomatik olarak kazınmıştır. Python programlama dili ile gerçekleştirilen bu işlemde, İstanbul'un belirli tarih aralıklarındaki hava durumu verileri saatlik dilimlere göre alınmıştır.[1,2]

B. API Entegrasyonu:

OpenWeather API kullanılarak, İstanbul için ek hava durumu verileri elde edilmiştir. API anahtarı oluşturularak, belirli zaman aralıklarına ait sıcaklık, nem ve rüzgar gibi meteorolojik parametreler çekilmiştir. Bu veriler de, JSON formatında saklanıp analiz sürecine dahil edilmiştir.[6,10]

C. Veri Formatlama ve Saklama:

Her iki kaynaktan (Meteostat ve OpenWeather API) gelen veriler **JSON** formatında düzenlenmiş ve **Python** aracılığıyla işlenmiştir. Son olarak, bu veriler analiz için **Excel** formatına dönüştürülerek saklanmıştır. Verilerin zaman serisi analizi ve sınıflandırma işlemleri için gerekli formatlama işlemleri yapılmıştır. Bu yöntemlerle elde edilen veriler, İstanbul'un hava durumu tahminleri için kullanılan modelin geliştirilmesine temeloluşturmuş ve tahmin sonuçlarının doğruluğunu artırmaya yardımcı olmuştur.[12]

4. Zaman Serisi

Zaman Serisinde Verilerin Zaman Dilimlerine Göre Ayırılması:

Zaman serisi verilerini analiz ederken, verilerin belirli zaman dilimlerine göre sınıflandırılması, analiz sonuçlarının daha ayrıntılı ve anlamlı olmasını sağlayabilir. Projemizde, elde edilen zaman serisi verilerini sabah, öğle, akşam ve gece olmak üzere dört farklı zaman dilimine ayırarak analiz ettik. Bu süreç, hava durumu tahminleri gibi zamanla değişen verilerin daha iyi yorumlanmasını sağlayarak mevsimsellik ve günlük trendlerin tespit edilmesine olanak tanır.

Bu zaman dilimlerini ayırırken şu saat aralıklarını kullandık:

- Sabah: 06:00 – 12:00
- Öğle: 12:00 – 18:00
- Akşam: 18:00 – 24:00
- Gece: 00:00 – 06:00

Zaman Serisinde Veri İşlenmesi:
Zaman serisi verilerini analiz ederken, eksik verilerin doldurulması, zaman serilerinin yeniden örneklenmesi, trend ve mevsimsellik bileşenlerinin ayrıştırılması gibi işlemler gerçekleştirilmiştir. Bu adımlar sonucunda veriler işlenmiş, grafikler oluşturulmuş, trend ve mevsimsellik bileşenleri analiz edilmiş ve eksik veriler lineer enterpolasyon yöntemiyle doldurulmuştur.
Zaman serilerinde **eksik verilerin doldurulması**, veri analizinin bütünlüğünü korumak ve modelleme sürecinde tutarlı sonuçlar elde etmek için önemli bir adımdır. Eksik veriler analiz sonuçlarını bozabileceği için, bu eksik noktaların doğru şekilde doldurulması gerekir. Bu işlemi yapmak için enterpolasyon yöntemi kullanılabilir. [12,14]
Enterpolasyon, mevcut veriler arasındaki boşlukları doldurmak için kullanılan bir tekniktir. Eksik olan veri noktalarını, verinin düzenine ve yapısına uygun şekilde tahmin eder. En yaygın kullanılan yöntemlerden biri **lineer enterpolasyon**dur.
Zaman serisindeki eksik verileri doldurmak için **lineer enterpolasyon** yöntemi kullanıldı. Bu yöntem, eksik olan noktaları doğrusal olarak komşu veri noktalarına göre hesaplayarak tahmin eder. Bu işlem sırasında, Python'daki **pandas** kütüphanesi kullanıldı ve `interpolate()` fonksiyonu ile eksik veriler lineer enterpolasyon yöntemiyle dolduruldu.

Mevsimsel ve Trendsel Bileşenlerin Ayrıştırılması:

Mevsimsel ve trendsel bileşenlerin ayrıştırılması, zaman serisi analizinde verinin uzun vadeli eğilimlerini (trend), düzenli olarak tekrar eden döngüsel değişimlerini (mevsimsellik), ve rastgele değişimleri (artık) daha net bir şekilde anlamak için kullanılan bir tekniktir. Bu işlem, zaman serisinin her bileşeninin etkisini ayrı ayrı değerlendirmeye yardımcı olur ve böylece gelecekteki değerlerin daha doğru bir şekilde tahmin edilmesini sağlar. Verinin daha iyi anlaşılması, karar alma süreçlerinde önemli rol oynar.

Zaman serisindeki farklı bileşenleri ayırmak, her bir bileşenin modelin tahminlerindeki etkisini netleştirir. Bu sayede, modelin doğruluğu artar ve daha anlamlı sonuçlar elde edilir. Örneğin, bir zaman serisinin yalnızca mevsimsel etkilerini incelemek, sadece

yıllık döngüleri göz önünde bulundurarak doğru tahminlerde bulunmayı sağlar.

seasonal_decompose fonksiyonu, zaman serisi verilerini analiz etmek ve bu verileri trend, mevsimsellik (seasonal) ve artık (residual) bileşenlerine ayırmak için yaygın olarak kullanılır. Bu fonksiyon, verinin temel bileşenlerini ortaya çıkararak her birinin zaman içindeki etkilerini görselleştirir.

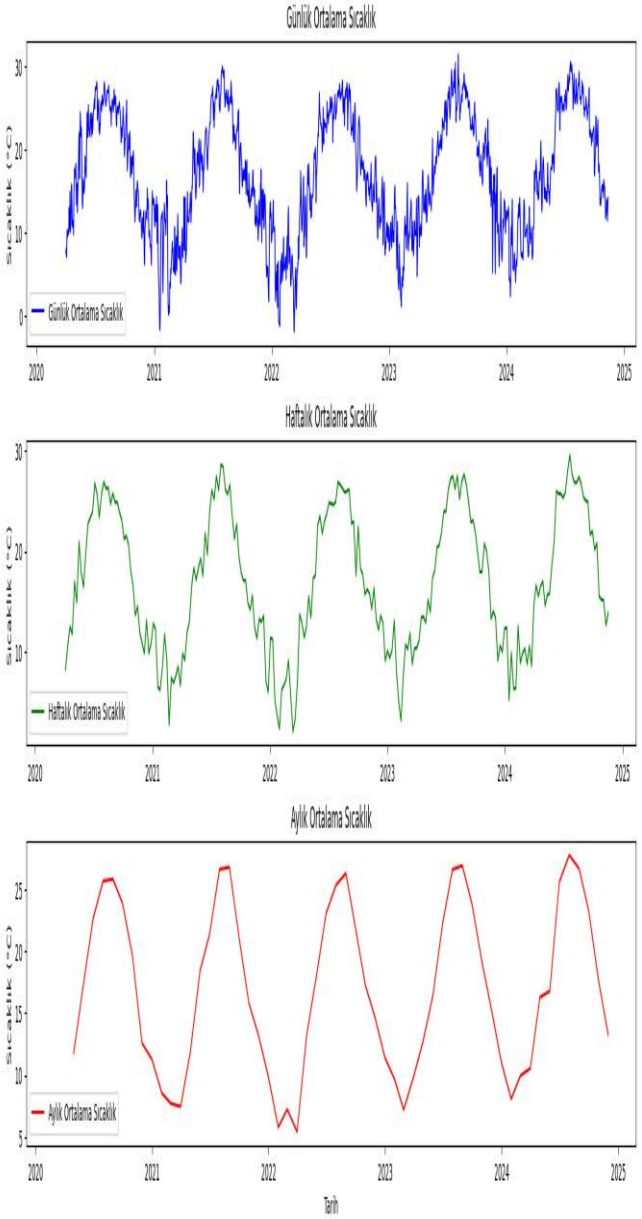
Trend bileşeni, zaman serisindeki uzun vadeli genel eğilimleri yansıtır. Bu bileşen, veri setinde yükselme veya düşme yönünde sürekli bir hareket olup, belirli bir zaman diliminde gözlemlenen eğilimlerin analizine olanak tanır. Örneğin, bir şirketin yıllık gelirlerinde görülen sürekli artış veya azalma, bir trendi gösterir.

Mevsimsellik bileşeni, belirli periyotlarla düzenli olarak tekrarlanan değişimleri ifade eder. Bu değişimler genellikle yıllık, aylık veya günlük gibi sabit zaman dilimlerine dayalı olarak ortaya çıkar. Örneğin, kışın sıcaklıkların düşmesi ya da yazın turizm sektöründeki yoğunlaşma gibi mevsimsel etkiler, mevsimselliğin tipik örneklerindendir.

Residual bileşeni: Zaman serisi analizlerinde, veriyi açıklamak için kullanılan modelin tahmin ettiği değerler ile gerçek gözlemler arasındaki fark residual (artık) bileşenini oluşturur. Trend ve mevsimsel bileşenler, verilerin uzun vadeli değişimlerini ve belirli periyotlarla (örneğin mevsimsel) tekrar eden desenlerini açıklarken, residual bileşeni, bu faktörler dışında kalan, yani modelin tahmin edemediği rastgele değişimleri ve hataları yansıtır.

Residual bileşeni, genellikle rastgele dağılan bir gürültü olarak kabul edilir ve bu nedenle modelin tahminlerinde açıklanamayan varyansı gösterir. Eğer model doğru bir şekilde kurulmuşsa, residual bileşeninin ortalaması sıfır olmalı ve düzenli bir yapı içermemelidir. Aksi takdirde, modelin eksik veya hatalı olduğunu, daha karmaşık bileşenlerin veya ilişkilerin göz ardı edildiğini gösterebilir.

Zaman serisi analizinde residual bileşeni, modelin doğruluğunu değerlendirmek için önemli bir araçtır. Analiz edilen residual'lar, modelin gelecekteki veri tahminlerinde kullanılabilirliğini etkileyebilir. Bu nedenle, residual bileşeninin düzgün bir şekilde analiz edilmesi ve olası hata kaynaklarının ortadan kaldırılması gerekmektedir.[12-14]



İstanbul için bu grafiklerde görülen döngüsel yapı, şehrin dört mevsimi belirgin olarak yaşadığını ve sıcaklıkların mevsimlere göre düzenli bir şekilde değiştiğini gösteriyor. Mevsimsel değişiklikler, şehirdeki sıcaklıkların yıllık döngülerle net bir şekilde tekrarladığını ortaya koyuyor. Özellikle yaz aylarında sıcaklıklar yüksek, kış aylarında ise oldukça düşük seyretmekte. Bu grafikteki düzenli döngü, İstanbul'un ikliminin yıllar arasında çok büyük bir değişiklik göstermediğini ve sıcaklıkların yıllık bazda tutarlı bir şekilde tekrar ettiğini işaret ediyor. Mevsimsel değişimlerin düzenliliği, İstanbul'daki iklimin uzun vadede stabil olduğunu ve sıcaklıkların yıllık döngüye bağlı olarak öngörülebilir bir şekilde değiştiğini gösteriyor. İstanbul'da mevsim geçiş dönemlerinde (özellikle ilkbahar ve sonbahar) sıcaklık değişimleri daha yavaş bir artış veya azalış eğilimi sergilerken, yaz ve kış dönemleri sıcaklık açısından daha kararlı bir döneme işaret ediyor. Yazın sıcaklıklar belirgin bir şekilde yükselirken, kışın ise sıcaklıklar keskin bir şekilde düşmektedir. Bu düzenli yapı, İstanbul'un iklimsel yapısının istikrarlı olduğunu ve bu eğilimlerin uzun yıllar boyunca süregeldiğini vurgulamaktadır. Bu döngüler, şehri ziyaret edenlerin ve orada yaşayanların iklimsel değişimleri daha iyi anlamalarına yardımcı olabilecek önemli veriler sunmaktadır. [12-14]

5.Kaynakca

1. Selenium Kulannımı ve kütühpaneler nasıl yaptık: https://www.selenium.dev/documentation/
2. Selenium web cromhe driver bağlanarak nasıl açılır: https://medium.com/kodluyoruz/selenium-webdriver-ile-test-senaryosu-107ad1f99229
3. Temel Selenium Komutları ve İşlevleri: Web Otomasyonunun: https://muhammedtalhacevik.medium.com/temel-selenium-komutlar%C4%B1-ve-i%CC%87%C5%9Flevleri-web-otomasyonunun-temelleri-a6be9066f64d
4. Python kütüphaneleri Pandas ,Tutorial, *W3Schools*. : https://www.w3schools.com/python/pandas/default.asp
5. "Python Selenium ile Web Otomasyon Testi Çalıştırma," *BrowserStack*. Erişim: https://www-browserstack-com.translate.goog/guide/python-selenium-to-run-web-automation-test?_x_tr_sl=en&_x_tr_tl=tr&_x_tr_hl=tr&_x_tr_pto=wa
6. *Meteostat*. Erişim: https://meteostat.net/en/
7. "Selenium'da Beklemeler," *Selenium-Python Dokümantasyonu*. Erişim: https://selenium-python.readthedocs.io/waits.html
8. "Maven Repository." Erişim: https://mvnrepository.com/
9. “Numpy Giriş,” *W3Schools*. Erişim: https://www.w3schools.com/python/numpy/numpy_intro.asp
10. "REST API," *Dev.to*. Erişim: https://dev.to/aciklab/rest-api-e26
11. M. Agrawal, "Selenium Kullanarak Web Scraping Nasıl Yapılır: Örnek ile," *Towards Data Science*, 20 Şubat 2022. Erişim: https://towardsdatascience.com/how-to-use-selenium-to-web-scrape-with-example-80f9b23a843a
12. "Verileri Dosyaya Güvenli Bir Biçimde Saklama," *W3Schools*. Erişim: https://www.w3schools.com/python/python_json.asp
13. "Matplotlib Time Series Line Plot," *DataCamp*. Erişim: https://www.datacamp.com/tutorial/matplotlib-time-series-line-plot
14. M. Günak, "Python ile Zaman Serileri Tahmin Yöntemleri," *Medium*, 12-May-2020. Erişim: https://medium.com/@mervegunak/python-ile-zaman-serileri-tahmin-y%C3%B6ntemleri-4eeb784d4562