ÖNSÖZ

Deprem dünyanın oluşum sürecine eşlik eden ve varlığını hiçbir zaman kaybetmeyecek gerçekçi bir oluşumdur. Farklı farklı zaman dilimlerinde insanoğluna varlığını hissettiren deprem çoğrafi konumu ile Türkiyeye de kucak açmaktadır. Deprem; oluşumu itibari ile yer kabuğundaki enerji birikiminin aniden serbest kalmasıyla gerçkleşir.Bu süreç yer kabuğundaki farklı tektonik levhaların harket halinde olması ile lvhalar hareket ederkn , birbirlerine sürtünerek veya baskı yapark gerilim oluştururlar.bu gerilim yer kabuğunda biriken deformasyon(bükülme veya çatlama da denilebilir) nedeniyle zamanla artar. Levhakarın birbirine yaklaşması, uzaklaşması ya da kayması büyük miktarda enerji birikmesine yol açar.Bu enerji birikiminin gerçekleştiği yerler yer kabuğundaki **faylar** olarak bilinen çatlaklardır. Faylar, yer kabuğundaki levhaların hareket ettiği kırılma ve çatlama noktalarıdır. Ve bu faylarda biriken enerji kritik bir noktaya ulaştığında,yer kabuğunda aniden bir kırılma meydana getirir.Bu kırılma sırasında,yer kabuğundaki gerilim serbest kalır ve sismik dalgalar olarak yayılır. Sismik dalgalar, yer yüzeyine doğru yayılır ve yer sarsıntısı olarak hissedilir.Türkiye konumu ve çoğrafi yapısı ile 485 diri fay hattı ile bir deprem ülkesi olarak varlığını sürdürmektedir ve insanoğlu bu kadar fay hattı sayısı ile yıllarca bu sarsıntılar ile karşı karşıya kalmş ve de bu sarsıntıların ağır sonuçlar doğurmasıyla yakınmışlardır.Peki sarsıntıların önüne geçilemese bile önceden önlem alamaz mıyız?

Yahya Kemal Beyatlı’nın da dediği gibi "Geçmişe bakarak, geleceği biçimle!" dizesindn yola çıkarak bu konuyu ele almamız gerekirse geçmişte gerçkleşen depremlerin istatistiksel verileri ile gelecekteki olası depremleri öngörebilir ve depremin olmasına engel olunamasa bile önceden önlemler alınarak insanoğlu için doğuracağı oluşumlar azaltılabilir.

GİRİŞ

Deprem insanoğlunu yaşamı boyunca sayısız kez etkilemiş ve derin acılara derin kayıplara sebep olmuştur.Hep insanoğlu önlem almayı ertelemiş ve olası ihtiamlleri göz ardı etmşştir.Hazırsız yakalanılan depremleri azaltmak ve olabiliritesi olan şeyleri tespit etmek için çeşitli yollar vardır.Deprem ile ilgili hali hazırda oluşan depremler hakkında bilgi var olmakla birlikte bu bilgilerden yola çıkarak tahmin konusunda insanoğlunun bilgi alabileceği aktif bi sistem bulunmamktadır.Deprem her an kapı çalabilecek bir tehlike olarak yaşamında insanoğluna eşlik ederken bu konuda daha bilgili ve öngörebilir olmak gerekir. **jeofizikçiler**, **yer bilimciler**, **deprem mühendisleri**, **sismologlar**, **deprem tahmincileri** ve **yerbilimciler gibi kişiler bu alanlarda araştırmalar yapıp bilgi birikimlerini insanoğluna iletmesinin yanında insanoğlunun olası tahminleri artk hayatımızda deprem kadar gerçk olan yapay zeka ve internet aracışığı ile olası tahminleri bi de bu taraftan ele alabilmk mümkündür.**

**İnternet geniş bir yelpazeyi kapsayan uçsuz bucaksız bir veri ve bilgi dağarcığını sunmaktadır.Deprem konusunda da veriyi bilgiye çevirme görevini proje aracılığyla ilerletebilir.**

**YÖNTEM**

**KULLANILAN KÜTÜPHANELER:**.

* **PyQt5**: Masaüstü uygulamaları geliştirmek için kullanılan GUI framework'ü. Kullanıcı arayüzü elemanları (butonlar, etiketler, grafikler vb.) ile çalışmak için gereklidir.
* **Dash** ve **Plotly**: Veri görselleştirme ve interaktif grafikler için kullanılır.
* **Requests**:
* Bu kütüphane, HTTP üzerinden veri çekmek için kullanılır. Web sitelerinden veri almak (web scraping) veya bir API'den veri çekmek için HTTP istekleri (GET, POST, vb.) göndermemizi sağlar.
* Bu kodda, özellikle web sitesinden depremlerle ilgili veriyi almak için kullanılmışTIR.
* **Pandas**:
* Python'da veri işleme ve analiz için en çok kullanılan kütüphanelerden biridir.
* pandas ile tablo yapısındaki veriler DataFrame formatında tutulabilir ve üzerinde analiz, filtreleme, temizleme gibi işlemler yapılabilir.
* Bu kodda, çekilen deprem verilerini bir DataFrame içine koyarak, kolay analiz edilebilir bir yapıya getirmek için kullanılmıştır.
* **Bs4 (BeautifulSoup)**:
* BeautifulSoup (bs4 modülünden) HTML ve XML dosyalarını işlemek için kullanılır.
* Bir web sayfasının kaynak kodundan istenen bilgileri ayıklamak, örneğin belirli HTML etiketleri içindeki metinleri almak için kullanılır.
* Bu kodda, requests ile çekilen web sayfasının HTML yapısını analiz ederek, ihtiyaç olan deprem verilerini ayıklamak için kullanılmıştr.
* **matplotlib.pyplot (plt)**:
* matplotlib kütüphanesi, Python'da veri görselleştirmek için kullanılan en popüler kütüphanelerden biridir.
* pyplot, bu kütüphanenin grafik çizme, eksen ekleme, başlık koyma, vb. temel işlevleri içeren bir alt modülüdür.
* Bu kodda, deprem verilerini çizgi grafiği veya başka tür grafikler şeklinde görselleştirmek için kullanılmıştır.
* **sklearn.ensemble.RandomForestRegressor**:
* RandomForestRegressor, scikit-learn kütüphanesinin bir parçasıdır ve regresyon problemleri için kullanılan bir makine öğrenimi algoritmasıdır.
* Random Forest, birçok karar ağacından oluşan bir ansambl algoritmasıdır. Her ağaç veriden bağımsız olarak öğrenir ve regresyon için bu ağaçların ortalama tahmini alınır.
* Bu kodda, geçmiş deprem verilerinden hareketle gelecekteki büyüklükleri veya başka deprem özelliklerini tahmin etmek için kullanılmıştır.
* **sklearn.metrics.mean\_squared\_error**:
* Bu modül, makine öğrenimi modellerinin performansını değerlendirmek için kullanılan mean squared error (MSE, ortalama kare hatası) metriğini sağlar.
* mean\_squared\_error, modelin tahmin ettiği değerler ile gerçek değerler arasındaki farkların karesinin ortalamasını hesaplayarak, modelin hata miktarını ölçer. Küçük bir değer, daha iyi bir tahmin performansı anlamına gelir.
* Bu kodda, RandomForestRegressor modelinin tahmin doğruluğunu değerlendirmek için kullanılmıştır.
* **sys**:
* Bu Python standart kütüphanesi, sistemle ilgili işlevleri içerir.
* Örneğin, programın kapanmasını kontrol etme (sys.exit()), komut satırı argümanlarını okuma (sys.argv), veya sistem yolu ayarlarını yapma gibi işlemler için kullanılır.
* **dash**:
* Dash, Python tabanlı web uygulamaları geliştirmek için kullanılan bir çerçevedir.
* Veri görselleştirme ve analiz projelerinde interaktif kullanıcı arayüzleri oluşturmak için kullanılır. Özellikle veri bilimciler ve analistler tarafından veri panelleri ve grafiksel gösterimlerle etkileşimli uygulamalar yapmak için tercih edilir.
* dash ile HTML ve grafikleri kolayca tanımlanabilir, ayrıca olay tabanlı kullanıcı etkileşimleri yönetilebilir.
* **dash.dcc ve dash.html**:
* dcc (Dash Core Components), Dash ile kullanılabilen grafik, dropdown, slider gibi çeşitli bileşenleri içerir.
* html modülü, HTML bileşenlerini (div, h1, p gibi) tanımlamak için kullanılır. Bu sayede Dash uygulamalarında HTML yapısını kolayca oluşturabiliriz.
* Bu kodda, Dash ile bir web arayüzü oluşturulurken kullanılacak görsel ve yapısal bileşenler için bu modüllere ihtiyaç duyulmaktadır.
* **plotly.express (px)**:
* plotly.express, veri görselleştirme kütüphanesi olan Plotly'nin basit arayüzüdür.
* Grafiklerin kolayca oluşturulmasına imkan tanır ve Dash ile birlikte kullanıldığında grafiklerin etkileşimli hale gelmesini sağlar.
* Bu kodda, veri setine dayalı grafikler oluşturmak için kullanılmış olabilir.
* **pandas (pd)**:
* pandas, Python'da veri işleme ve analiz için kullanılan temel kütüphanelerden biridir.
* Tablo verilerini saklama, işleme ve analiz etme gibi işlemler için kullanılır. Veriler bir DataFrame nesnesinde tutulur ve kolayca manipüle edilebilir.
* Bu kodda, verilerin saklanması, analiz edilmesi ve grafikler oluşturulması amacıyla kullanılmış olabilir.
* **PyQt5.QtCore ve PyQt5.QtWidgets**:
* PyQt5, Python için GUI (Grafik Kullanıcı Arayüzü) kütüphanesidir ve Qt GUI çerçevesinin bir arayüzüdür.
* QtCore modülü, temel Qt işlevlerini sağlar ve PyQt5 uygulamalarında zamanlayıcılar, sinyaller, olaylar gibi temel işlevler için kullanılır.
* QtWidgets modülü ise butonlar, pencereler, layout'lar gibi grafiksel bileşenleri içerir. Bu bileşenler, kullanıcı arayüzü tasarımı için gereklidir.
* **QMainWindow, QVBoxLayout, QWidget**:
* Bu sınıflar PyQt5 içinde kullanıcı arayüzü oluşturmak için kullanılır.
* QMainWindow, ana pencereyi temsil eder. Menü çubuğu, araç çubuğu, durum çubuğu gibi bileşenler eklenebilir.
* QVBoxLayout, dikey bir düzen (layout) sağlar. Bileşenleri dikey sırayla düzenlemek için kullanılır.
* QWidget, temel bir bileşen olarak kullanılır ve diğer bileşenler üzerine yerleştirilir.
* **PyQt5.QtWebEngineWidgets.QWebEngineView**:
* QWebEngineView, PyQt5 uygulamasında bir web tarayıcı bileşeni sağlar. Web sayfalarını veya Dash gibi bir web tabanlı uygulamayı bir GUI uygulaması içinde görüntülemek için kullanılır.
* Bu bileşen, Dash uygulamasını PyQt5 arayüzü içinde görüntülemek amacıyla kullanılabilir.

**Hali hazırda kandilli rasathanesinde bulunan depremin anlık değerlerinin düştüğü veriler ile proje yürütülecektir.**

1. **Deprem Verisi Çekme (HTTP İsteği)**

Deprem verilerini almak için, **requests** kütüphanesi kullanılmış ve **Koeri (Boğaziçi Üniversitesi Kandilli Rasathanesi)** web sitesinden deprem verilerine ulaşılmaya çalışılmıştır. URL olarak şu adres kullanılmıştır: “http://www.koeri.boun.edu.tr/scripts/lst1.asp“

Bu URL, Boğaziçi Üniversitesi Kandilli Rasathanesi'nin internet sitesinde, Türkiye'deki son depremleri içeren bir sayfadır. Bu sayfa dinamik olarak güncellenen verileri içermektedir.

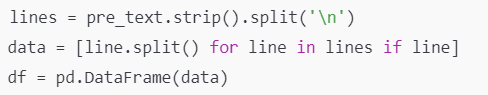
* **requests.get(url)** fonksiyonu ile web sayfasının HTML içeriği elde edilmiştir.
* **response.raise\_for\_status()** ile sayfanın başarılı bir şekilde çekilip çekilmediği kontrol edilmiştir. Eğer bir hata oluşursa, örneğin sayfa erişilemezse, bu fonksiyon bir hata fırlatır.
* HTML içeriği elde edildikten sonra, **BeautifulSoup** kullanılarak HTML parse edilmiştir. Bu, sayfanın yapısını anlamamıza ve veriye ulaşmamıza olanak tanır.



Web sayfasında deprem verileri genellikle **<pre>** etiketleri içinde yer almaktadır. Bu etiketler, verilerin düzenli bir şekilde görüntülenmesini sağlar (genellikle düz metin formatında).



* **pre\_tags[0].get\_text()** komutuyla, ilk <pre> etiketinin içeriği alınır ve deprem verilerini içeren metin elde edilir.
* Bu metin, satırlara ayrılarak **split('\n')** ile her bir satır bir liste elemanı haline getirilir.
* **line.split()** komutuyla her satırdaki boşluklarla ayrılarak veri sütunlarına ayrılır.
* Elde edilen veriler bir **DataFrame**'e dönüştürülür (pandas kullanılarak).

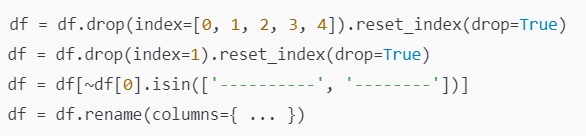


Veriler başarılı bir şekilde çekildikten sonra, verilerin düzenlenmiş hali **deprem.csv** dosyasına kaydedilir. Bu, gelecekteki kullanım için verilerin bir dosyada saklanmasını sağlar.



Deprem verisi, genellikle HTML sayfasında başlıklar ve gereksiz boş satırlar içerdiğinden, düzenleme (preprocessing) işlemi yapılmıştır:

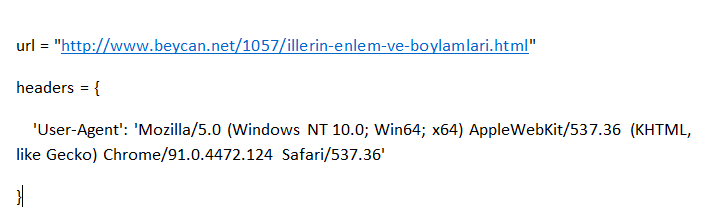
* İlk birkaç satır (başlıklar, açıklamalar) silinmiştir.
* "----------" ve "--------" gibi ayırıcı satırlar da çıkarılmıştır.
* Veri sütunlarına anlamlı isimler verilmiştir.



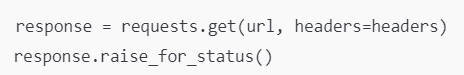
Deprem verileri başarılı bir şekilde web sayfasından çekilmiştir. Web scraping yapılarak, sayfanın HTML içeriği parse edilmiştir ve deprem verileri bir DataFrame'e dönüştürülüp daha sonra CSV dosyasına kaydedilmiştir. Bu, veri analizi veya modelleme için kullanılabilir hale gelmiştir.

Aynı zamanda aynı deprem verilerinin çekildiği gibi şehirlerin enlem ve boylam bilgilerini de <http://www.beycan.net/1057/illerin-enlem-ve-boylamlari.html> uzantılı sayfadan bir fonksiyo içinde çekerek proje veri dağarcığına bir dataframe olark dahil etmektedir.

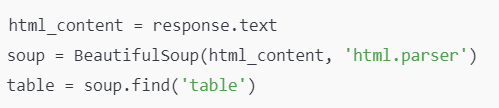
* url: Veri çekilmek istenen web sayfasının URL'sini belirtir. Bu URL, şehirlerin enlem ve boylam bilgilerini içeren bir tabloya sahip bir sayfadır.
* headers: HTTP isteğiyle birlikte gönderilen başlık bilgilerini içerir. Bu başlıklar, web sunucusuna isteği gönderenin bir tarayıcı olduğunu bildirir. User-Agent bilgisi, genellikle web sayfalarının "bot"lar (yani, otomatik veri çekme araçları) tarafından erişilmeyi engellemeleri için kullanılır. Bu başlık, botların engellenmesini önler.

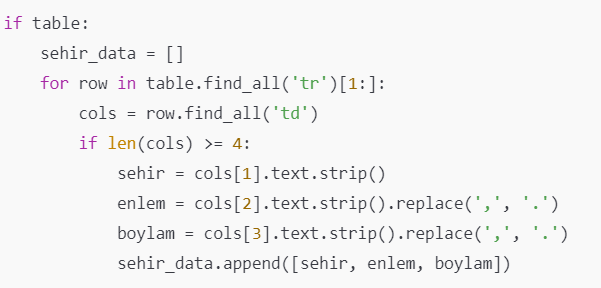


* requests.get(url, headers=headers): Web sayfasına HTTP GET isteği gönderilir. Bu istek, URL'nin içeriğini almak için kullanılır. Başlıklar da burada belirtilir.
* response.raise\_for\_status(): Eğer HTTP isteği başarısız olursa (örneğin, sayfa bulunamazsa veya sunucu hatası meydana gelirse), bu fonksiyon bir hata fırlatır. Böylece hatalı istekler kolayca tespit edilebilir.



* html\_content = response.text: Sunucudan dönen yanıtın içeriği (text olarak) alınır. Bu içerik, HTML biçimindedir ve BeautifulSoup ile işlenecektir.
* soup = BeautifulSoup(html\_content, 'html.parser'): html.parser kullanarak, HTML içeriği BeautifulSoup ile işlenebilir bir formata dönüştürülür. BeautifulSoup, HTML belgelerini analiz etmek ve bunlardan veri çekmek için yaygın olarak kullanılan bir Python kütüphanesidir.
* table = soup.find('table'): Sayfadaki ilk <table> HTML etiketini arar ve onu table değişkenine atar. Bu etiket, şehirlerin enlem ve boylam bilgilerini içeren tablonun bulunduğu HTML etiketidir.



* if table:: Eğer sayfada <table> etiketi bulunduysa (yani tablonun varlığı doğrulandıysa), işlem devam eder.
* sehir\_data = []: Bu liste, şehirlerin adı, enlem ve boylam verilerini tutacaktır.
* for row in table.find\_all('tr')[1:]:: table.find\_all('tr') ile tablo içindeki tüm satırlar (<tr> etiketleri) bulunur. table.find\_all('tr')[1:] ile ilk satır (başlık satırı) hariç tutulur. Bu döngü, her bir veri satırını işlemeye başlar.
* cols = row.find\_all('td'): Her bir satırdaki hücreleri (<td> etiketleri) alır. Bu hücrelerdeki veriler, şehir adı, enlem, boylam gibi bilgileri içerir.
* if len(cols) >= 4:: Eğer satırda en az 4 hücre varsa (bazı satırlarda eksik veri olabileceği için), bu satır işleme alınır.
* sehir = cols[1].text.strip(): Şehir adı, satırdaki ikinci hücrede bulunur (cols[1]). .text ile metin içerik alınır, .strip() ile baştaki ve sondaki boşluklar kaldırılır.
* enlem = cols[2].text.strip().replace(',', '.'): Enlem değeri, üçüncü hücrede bulunur (cols[2]). Bu değerdeki virgül (,) noktaya (.) dönüştürülür. Bu, bazı sayfalarda enlem ve boylam verilerinin ondalıklı biçimde virgül ile yazılması nedeniyle yapılır.
* boylam = cols[3].text.strip().replace(',', '.'): Boylam değeri, dördüncü hücrede yer alır ve benzer şekilde işlenir.
* sehir\_data.append([sehir, enlem, boylam]): Şehir adı, enlem ve boylam bilgileri bir liste olarak sehir\_data listesine eklenir.
* df = pd.DataFrame(sehir\_data, columns=['Şehir', 'Enlem', 'Boylam']): Elde edilen şehir verileri, pandas DataFrame formatına dönüştürülür. sehir\_data listesi, her bir satırda şehir adı, enlem ve boylam bilgilerini tutar. columns parametresi, bu sütunlara uygun isimler verir.
* df.to\_csv('sehirler\_enlem\_boylam.csv', index=False): Bu veriler, sehirler\_enlem\_boylam.csv adında bir CSV dosyasına yazılır. index=False parametresi, DataFrame’in indeksinin dosyaya yazılmamasını sağlar.

Bu fonksiyon, bir web sayfasından (Beycan.net'teki bir sayfadan) şehirlerin enlem ve boylam bilgilerini çeker. HTML sayfasındaki bir tabloyu parse eder, ilgili verileri alır ve bu verileri pandas DataFrame formatında düzenler. Son olarak, bu verileri bir CSV dosyasına kaydeder. Eğer tablo bulunamazsa ya da istek sırasında bir hata olursa, uygun hata mesajları yazdırılır.

Bir fonksiyon içeriisnde bir **makine öğrenmesi** modelini eğitmek için kullanılıyor ve deprem verilerini kullanarak **deprem büyüklüğünü (ML)** tahmin etmeyi amaçlayan bir model oluşturuyor. Kodun her bir kısmını adım adım inceleyelim:

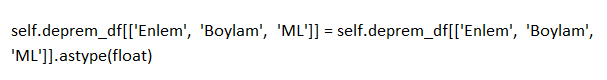
1.Deprem verisinin kontrolü :

Bu satır, self.deprem\_df veri çerçevesinin boş olup olmadığını kontrol eder. Eğer self.deprem\_df boş değilse (yani deprem verileri başarıyla yüklenmişse), model eğitme işlemi başlar.

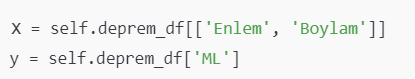


2.Veri tipi dönüştürme:

* astype(float): Bu satır, **Enlem**, **Boylam** ve **ML (deprem büyüklüğü)** sütunlarındaki verilerin tipini **float** (ondalık sayılar) olarak değiştirir.
* Bu işlem, özellikle veri setinde sayısal değerlerin **string (metin)** formatında olabilmesi durumunda gereklidir. RandomForestRegressor gibi makine öğrenmesi modelleri sayısal verilerle çalıştığı için, bu sütunların sayısal türde olması sağlanır.

3. **Özellik ve Hedef Değişkenlerin Seçimi**:

* **X**: Bu değişken, modelin **bağımsız değişkenlerini** (özelliklerini) temsil eder. Burada Enlem ve Boylam sütunları kullanılıyor, çünkü bu özellikler modelin, yani depremin büyüklüğünü tahmin etmek için kullanılan girdilerdir.
* **y**: Bu değişken, modelin **bağımlı değişkenini** (hedef değişkenini) temsil eder. Burada ML (deprem büyüklüğü) kullanılıyor, çünkü modelin tahmin etmeye çalıştığı değerdir.



**4.Random Forest Modelinin Oluşturulması ve Eğitilmesi**:

**RandomForestRegressor**: Bu, bir **regresyon** modelidir. **Random Forest** (Rastgele Orman) algoritması, birden fazla **karar ağacı** kullanarak tahmin yapar.

* n\_estimators=100: Bu parametre, modelin kaç adet **karar ağacı** (decision tree) kullanacağını belirtir. Burada 100 ağaç kullanılıyor.
* random\_state=42: Bu parametre, modelin rastgele süreçlerinde **tekrarlanabilirlik** sağlamak için kullanılır. Aynı veriyle her çalıştırıldığında aynı sonuçları alabilmek için bir başlangıç noktası belirler. 42 herhangi bir sabit sayı olabilir, ama genellikle belirli bir sayı seçilir (istatistiksel literatürden tanıdık bir örnek olan 42 sıklıkla kullanılır).

**5.Modelin Eğitilmesi (Fitting)**:

fit(): Bu metot, modelin eğitilmesini sağlar. X (Enlem ve Boylam) özelliklerini ve y (ML) hedef değişkenini kullanarak model öğrenme sürecini başlatır. Yani, model bu verilerle çalışarak, Enlem ve Boylam'a göre ML (büyüklük) tahminlerini yapmayı öğrenir.



6.. **Modeli Döndürme (Return)**:

* Bu satır, eğitilmiş modeli **geri döndürüyor**. Yani, bu fonksiyon çağrıldığında, **RandomForestRegressor** modelinin eğitilmiş hali geri döndürülür.
* Eğitilen model, daha sonra farklı girdiler (örneğin yeni şehirlerin enlem ve boylam bilgileri) ile **deprem büyüklüğü tahmini yapmak** için kullanılabilir.



Bu fonksiyon, **Random Forest** algoritmasını kullanarak **deprem büyüklüğünü (ML)** tahmin etmek için bir model eğitir. Model, **Enlem** ve **Boylam** özelliklerini kullanarak **ML**'yi (deprem büyüklüğü) tahmin etmeyi öğrenir. Eğitilen model daha sonra kullanıma sunulmak üzere geri döndürülür.

Buradaki temel amaç, depremin **konumu** (Enlem, Boylam) ile **büyüklüğünü** (ML) tahmin edebilmek. Bu tür bir model, depremlerin gelecekteki büyüklüklerini tahmin etmek veya mevcut verilerle analiz yapmak için kullanılabilir.

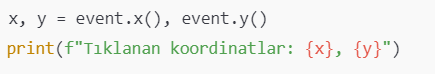
1.. **Mouse Press Event (Fare Tıklama Olayı)**:

* mousePressEvent fonksiyonu, PyQt5'te fare tıklama olaylarını yakalamak için kullanılır.
* event: QMouseEvent: Bu, fare tıklama olayını temsil eder. Olayın (örneğin tıklama) özelliklerine erişmek için bu nesne kullanılır.



2. **Tıklanan Koordinatları Almak**:

* event.x() ve event.y() metotları, fare tıklamasının **x** ve **y** koordinatlarını alır.
* Bu koordinatlar, tıklanan yerin ekran üzerindeki (veya pencere içindeki) konumunu belirler.



3. **Harita Üzerindeki Şehirlerin Koordinatlarıyla Karşılaştırma**:

* self.city\_map: Bu, her şehre ait koordinatları ve harita üzerindeki konumunu tutan bir liste veya dizi. Her şehir için (şehir adı, x, y, genişlik, yükseklik) bilgilerini içerir.
* city, cx, cy, width, height: Bu, her şehir için sırasıyla şehir adı, şehir merkezinin x ve y koordinatları, şehir alanının genişlik ve yüksekliğidir.



Burada tıklanan **(x, y)** koordinatları ile şehirlerin alanları karşılaştırılıyor.

* (cx - width <= x <= cx + width) ve (cy - height <= y <= cy + height) ile, tıklamanın şehir alanının içinde olup olmadığını kontrol eder.
* Bu kont



4. **Tıklanan Şehri Kullanıcıya Bildirmek**:

* Eğer tıklanan yer bir şehir alanına denk geldiyse, kullanıcıya bir bilgi penceresi (QMessageBox) gösterilir. Bu pencere, tıklanan şehri kullanıcıya bildirir.
* Aynı zamanda print() fonksiyonu ile konsola da tıklanan şehir yazdırılır.

5. **Şehir Adını Büyük Harfe Çevirme ve Veri Kontrolü**:

Şehir adı büyük harfe dönüştürülür, çünkü şehir isimlerinin karşılaştırılması yapılırken tutarlılık sağlanmak isteniyor.

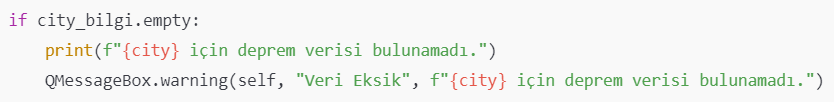


* self.sehir\_df: Şehirlerin bilgilerini (enlem, boylam) içeren bir veri çerçevesidir.
* Bu satırda, tıklanan şehrin adı (city\_name) ile **şehirin veri çerçevesindeki** (DataFrame) adı karşılaştırılır. Eğer şehir verisi bulunursa, şehirle ilgili enlem ve boylam bilgileri alınır.



6. **Şehir Verisi Bulunamazsa Uyarı Gösterme**:

Eğer **city\_bilgi** boşsa (empty), yani bu şehir için deprem verisi bulunamazsa, kullanıcıya bir **uyarı mesajı** gösterilir. Konsola da bir uyarı yazdırılır.



7. **Şehir Verisi Bulunursa Tahmin Yapma**:

Eğer şehir verisi varsa, o şehrin **enlem** ve **boylam** bilgileri alınır. Bu bilgiler, şehirle ilgili deprem tahmini yapmak için kullanılacaktır.



kullanici\_veri: Kullanıcının tıklamasıyla ilgili şehir bilgisini içeren bir **DataFrame** oluşturulur. Bu DataFrame yalnızca **Enlem** ve **Boylam** verilerini içerir. Bu veriler, **deprem büyüklüğü tahminini** yapacak olan modelin girdi verisi olacaktır.

* **trained\_model**: Daha önce eğitilmiş bir **makine öğrenmesi modelidir**. Bu model, **Enlem** ve **Boylam**'a göre **ML** (deprem büyüklüğü) tahmini yapmak için kullanılır.
* trained\_model.predict(kullanici\_veri): Bu fonksiyon, kullanici\_veri ile modelin tahmin yapmasını sağlar.



Eğer tahmin yapılabilirse, tahmin edilen deprem büyüklüğü (ML) konsola yazdırılır ve bir **bilgi mesajı** penceresi ile kullanıcıya gösterilir.



Bu kod parçası, kullanıcı harita üzerinde bir şehir üzerine tıkladığında:

* Tıklanan şehir ile ilgili verileri (enlem, boylam) alır,
* Bu verileri kullanarak bir **makine öğrenmesi modeli** ile deprem büyüklüğünü tahmin eder,
* Sonuçları kullanıcıya bildirir.

Bu, harita üzerinden interaktif bir şekilde deprem büyüklüğü tahmini yapmayı sağlayan bir fonksiyondur.

* QMainWindow: PyQt5'te temel pencere sınıfıdır. MyWindow sınıfı, ana pencerenin özelliklerini tanımlar.
* QWebEngineView: Bu sınıf, Dash uygulamasının görselleştirmelerini yerel PyQt5 penceresinde gösterebilmek için kullanılır. Dash uygulaması, bir web uygulaması olduğu için, bu sınıf Dash'in web sayfalarını görüntüler.
* QVBoxLayout: Pencere içinde elemanları dikey olarak düzenler.
* self.browser.setUrl(QUrl("http://127.0.0.1:8051")): Dash uygulaması burada çalıştırılacaktır ve bu URL, QWebEngineView'e yüklenir.

3. **Dash Uygulaması ve Deprem Verisi Çekme**:

* **requests** ile, belirtilen URL'den (Deprem verileri için Türkiye'nin KOERI sitesinden) veri çekilmektedir.
* **BeautifulSoup** ile, web sayfasındaki <pre> etiketleri içindeki veriler işlenip düzenlenir.
* Veriler pandas.DataFrame'e dönüştürülür ve CSV dosyasına yazılır.

4. **Veri Düzenleme**:

* Veri çektikten sonra, gereksiz satırlar (ilk birkaç satır) kaldırılır.
* Sütun isimleri daha anlamlı hale getirilir.
* 'Tarih' ve 'Saat' sütunları birleştirilerek yeni bir 'Tarih\_Saat' sütunu oluşturulur.
* Hatalı (eksik) tarih verileri temizlenir.

5. **Plotly ile Grafik Oluşturma**:

**Plotly Express** ile iki tür grafik oluşturuluyor:

1. **Zaman serisi (scatter grafiği)**: Deprem büyüklüklerinin zamanla değişimi.
2. **Bar grafiği**: Şehir bazında deprem büyüklüklerinin görselleştirilmesi.

6. **Dash Uygulaması Layout ve Callback**:

* **Dash Layout**: Dash uygulaması, kullanıcı arayüzü öğelerinin (grafikler, tarih aralığı seçici, vs.) yerleşimini tanımlar.
* **DatePickerRange**: Kullanıcının tarih aralığı seçmesine olanak tanır.
* **Graph**: Grafikler Dash tarafından render edilir.
* **Callback Fonksiyonu**: Kullanıcının tarih aralığını değiştirdiğinde, grafikleri yeniden çizmek için callback fonksiyonu çalıştırılır. Bu fonksiyon, kullanıcı seçimine göre verileri filtreler ve yeni grafikler oluşturur

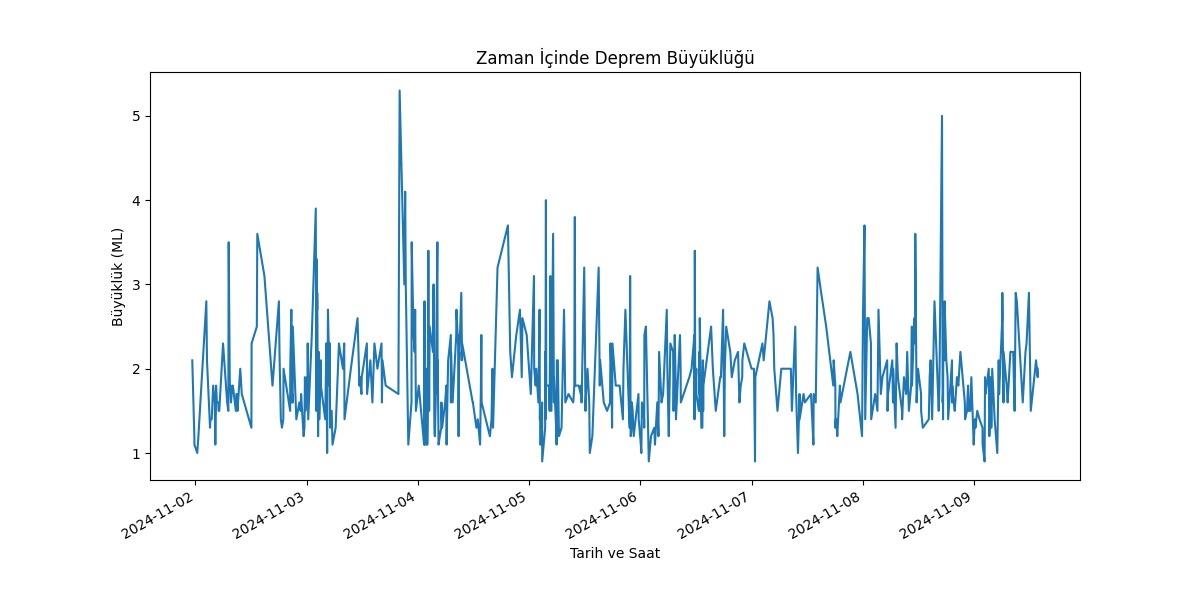


7. **Dash Sunucusunu Başlatmak ve Web Görüntülemesi**:

* Dash sunucusu, localhost:8051 adresinde çalıştırılır.
* QWebEngineView, PyQt5 penceresinde Dash uygulamasını görüntüler. Yani, Dash uygulaması bir web tarayıcısı içerisinde yerel bir PyQt5 GUI penceresinde gösterilir.

Bu uygulama, bir **PyQt5** masaüstü penceresinde **Dash** ile interaktif grafikler gösteren bir uygulamadır. Kullanıcı bir tarih aralığı seçtikçe, **deprem verileri** görselleştirilir. Veri, Türkiye'deki deprem bilgilerini sağlayan bir web sitesinden çekilir ve **Plotly** ile görselleştirilir. **PyQt5** penceresi, **Dash** web uygulamasını yerel olarak görüntüler, böylece masaüstü uygulamanızda web tabanlı bir içerik barındırabilirsiniz.

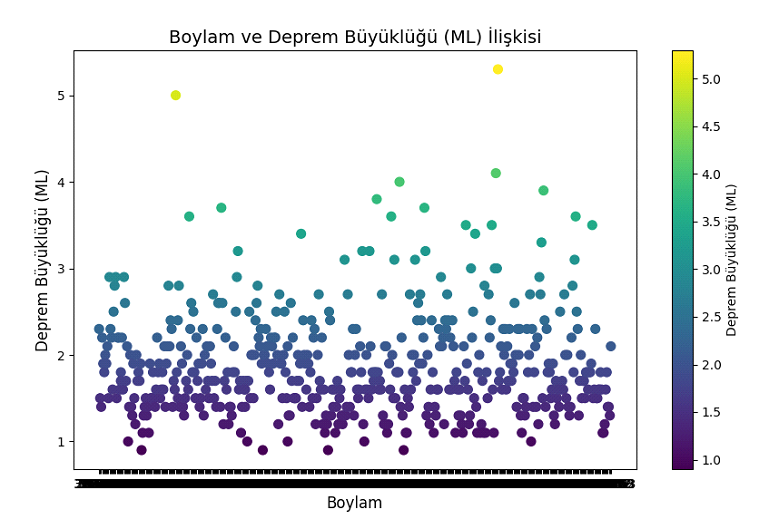
BULGULAR

Bu grafikte görülen düzenli dalgalanmalar, Türkiye gibi tektonik açıdan aktif bir bölgede olağan olan küçük ve orta şiddetli depremlerin zaman içindeki dağılımını göstermektedir. Çoğunlukla 1-3 ML arasında seyreden depremler, bölgede sürekli olarak meydana gelen sismik hareketlerin bir göstergesidir. Bu tür depremler genellikle küçük ölçekte olup insanlara pek hissettirilmez, ancak yer altındaki fay hatlarının aktif olduğunu ve sürekli enerji birikimi ve boşalması yaşandığını işaret etmektedir. Bu küçük depremler, büyük bir kırılmanın enerjisini biriktirmeden küçük salınımlar halinde boşaltılmasını sağlamaktadır ve bu şekilde büyük depremleri bir nebze geciktirebilmektedir.

3 Kasım ve 8 Kasım civarında görülen, büyüklüğü 5 ML’ye kadar çıkan birkaç deprem dikkat çekmektedir. Bu boyuttaki depremler, özellikle yüzeye yakın meydana gelirse hissedilebilir seviyededir ve bazı binalarda hafif hasara neden olabilmektedir. Bu daha büyük depremler, belirli bölgelerde fay hatlarının stres altında olduğunu ve daha büyük bir enerji birikimi yaşanabileceğini düşündürebilmektedir. Bu nedenle bu tür veriler deprem araştırmacıları ve yetkililer için önemli uyarı işaretlerini içermektedir.

Elde edilen bu tür veriler, bölgesel deprem hareketliliğinin takibi için son derece önemlidir. Düzenli izleme ile hangi fay hatlarının daha aktif olduğunu ve hangi bölgelerde enerji birikiminin arttığını gözlemlemek mümkün olmaktadır. Bunun yanında, bu verilerle uzun vadede belirli dönemlerde meydana gelen büyük depremlerin frekansını incelemek, olası gelecekteki büyük depremlerin zamanını tahmin etmeye yardımcı olabilmektedir.

Sonuç olarak, depremlerin zaman içindeki dağılımını incelemek, halkın ve yetkililerin deprem hazırlıkları konusunda daha bilinçli ve hazırlıklı olmasına katkıda bulunabilmektedir. Depremin kendisi öngörülemez olsa da, bu tür sismik veri analizleri olasılıkları anlamak ve uygun önlemleri almak açısından son derece değerlidir.

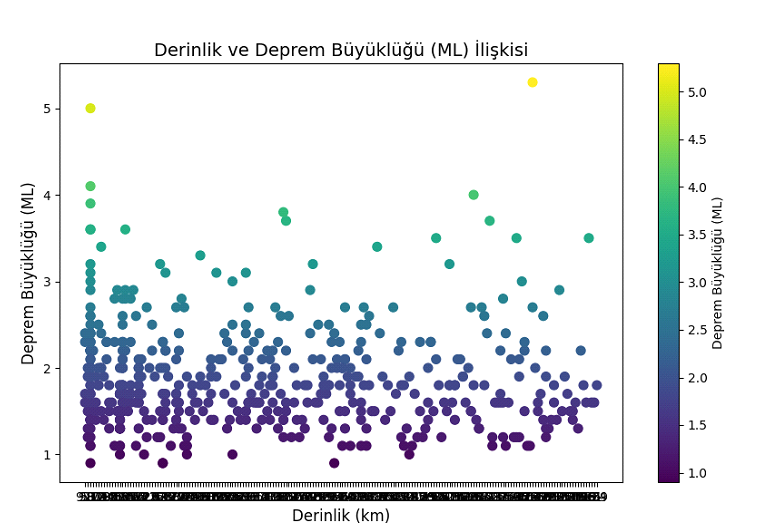
Bu grafikte, Türkiye'deki depremler için boylam ve deprem büyüklüğü (ML) arasındaki ilişki gösteriliyor. X ekseni boylamı, Y ekseni ise deprem büyüklüğünü ifade ediyor. Veri noktalarının renkleri, büyüklük derecelerine göre değişmekte; daha yüksek büyüklükteki depremler sarı tonlarında, daha düşük büyüklükteki depremler ise mor tonlarında gösterilmektedir. Sağdaki renk skalası, deprem büyüklüklerinin hangi renk ile temsil edildiğini daha iyi anlaşılmasını sağlamaktadır.

Büyüklük Dağılımı: Deprem büyüklüklerinin çoğu 1.0 ile 3.0 arasında yoğunlaşmış durumda, bu da genelde düşük şiddetli depremlerin yaygın olduğunu göstermektedir. 5.0 ve üzeri büyüklüğe sahip birkaç deprem noktası dikkat çekmektedir ancak bunlar az sayıda görülmektedir.

Boylam Farklılıkları: Türkiye'nin geniş boylam aralığı göz önünde bulundurulduğunda, bu grafikte belirli bir boylamın diğerlerinden daha fazla deprem ürettiğini gösteren belirgin bir odaklanma yoktur. Ancak, küçük depremler geniş bir alana yayılmış durumdadır.

Sismik Aktivitenin Coğrafi Dağılımı: Bu tür bir grafik, deprem büyüklüğü ile coğrafi koordinat arasında doğrudan bir ilişki olup olmadığını belirlemekten çok, depremlerin büyüklük dağılımını ve boylam eksenindeki yoğunluğunu anlamaya yönelik faydalı olmaktadır. Görülen dağılıma göre, Türkiye'deki depremler genellikle küçük ve orta şiddetlidir, ancak nadir de olsa büyük depremler meydana gelebilme ihtimali olmaktadır.

Sonuç olarak, grafikteki veri dağılımı, Türkiye'de depremlerin sürekli olarak meydana geldiğini ve farklı bölgelerde düşük şiddetli sismik aktivitelerin yaygın olduğunu göstermektedir.Büyük depremler ise seyrek fakat önemli olaylardır ve yerel olarak etkileri yüksek olabilir. Bu tür analizler, deprem risk haritalarının oluşturulmasına katkıda bulunabilir ve özellikle büyük depremlerin meydana gelme olasılığının daha yüksek olduğu bölgeleri belirlemek için önemli bir rol oynamaktadır.



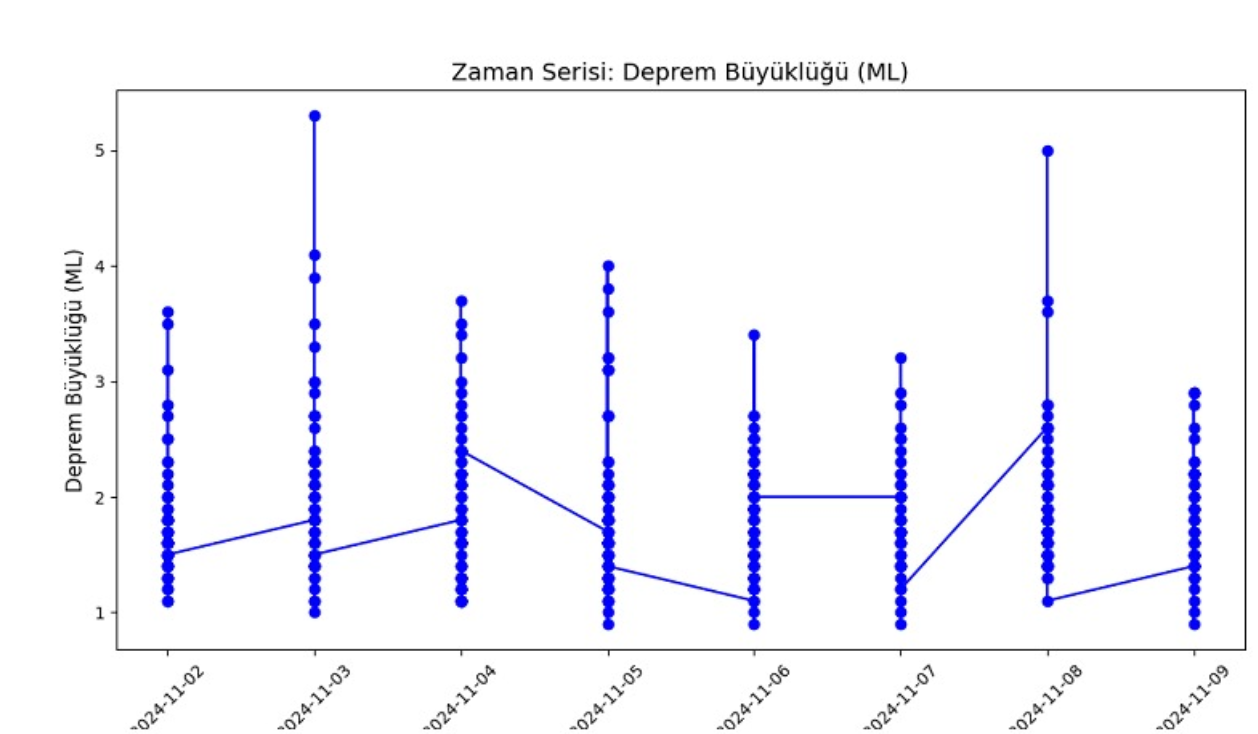
Bu grafikte, depremlerin derinlikleri ile büyüklükleri (ML) arasındaki ilişki gösterilmektedir. X ekseni derinliği (km cinsinden), Y ekseni ise deprem büyüklüğünü temsil etmektedir. Veri noktalarının renkleri, büyüklük değerlerine göre değişmektedir; daha yüksek büyüklükteki depremler sarı renge doğru, daha düşük büyüklükler ise mor renge doğru sıralanmıştır.

Derinlik Dağılımı: Depremler çoğunlukla yüzeye yakın, 10 km’nin altındaki derinliklerde yoğunlaşmış durumdadır. Bu sığ derinliklerdeki depremler, aynı büyüklükteki derin depremlerden daha fazla hissedilir ve yüzeyde daha fazla hasar yaratabilmektedir. Sığ derinliklerin bu kadar fazla olması, Türkiye’nin aktif fay hatlarının yüzeye yakın olduğunu ve bu fayların hareketlerinden dolayı sürekli küçük ve orta büyüklükte depremler meydana geldiğini göstermektedir.

Büyüklük ve Derinlik Arasındaki İlişki: Bu grafikte, büyüklük ile derinlik arasında belirgin bir ilişki olmadığını gözlemlemekteyiz. Yani, büyük depremler hem sığ hem de daha derin seviyelerde meydana gelebilir. 5.0 ve üzeri büyüklüğe sahip birkaç depremin sığ derinliklerde gerçekleştiği göze çarpmaktadır. Bu tür büyük ve sığ depremler, yıkıcı potansiyele sahip olabileceği için dikkat çekmektedir:

Genel Dağılım: Çoğu deprem 2.0 - 3.0 büyüklük aralığında bulunuyor, bu da sık rastlanan küçük sismik aktiviteleri temsil etmektedir. Büyük depremler az sayıda, ancak etkileri büyük olduğu için dikkate alınması gereken olaylardır.

Bu grafikteki veriler, Türkiye’nin sismik aktivitesinin büyük bir kısmının yüzeye yakın bölgelerde gerçekleştiğini ve düşük büyüklüklerde olduğunu göstermektedir. Bu tür analizler, deprem risk haritalarının oluşturulmasında önemli bir rol oynamaktadır. Özellikle sığ ve büyük depremlerden kaynaklanabilecek hasar risklerini belirlemek için faydalıdır. Bu tür görseller, hem akademik araştırmalar hem de toplumun bilgilendirilmesi için oldukça önemlidir.



Bu grafikte, belirli bir tarih aralığında kaydedilen depremlerin büyüklükleri (ML cinsinden) zaman serisi olarak gösterilmektedir. Yatay eksen, belirli günleri (tarihleri) temsil ederken, dikey eksen depremlerin büyüklüğünü göstermektedir. Her tarih için depremlerin büyüklüğü mavi noktalarla belirtilmiş ve bu noktalar birleştirilerek bir çizgi oluşturulmuştur.

Büyüklük Dağılımı: Grafikteki depremler genellikle 1 ila 5 büyüklük arasında değişiklik göstermektedir. Büyüklükteki ani sıçramalar (örneğin, 2 Kasım ve 8 Kasım tarihleri) dikkat çekici ve bu tarihlerde daha güçlü depremler kaydedilmiştir.

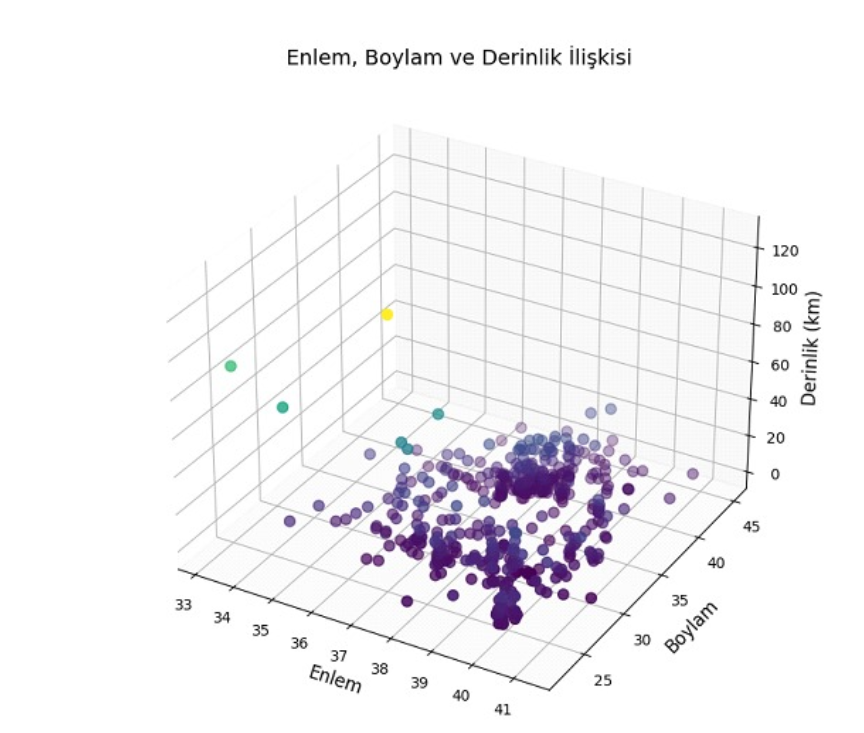
Düzenlilik: Deprem büyüklüklerinde belirli bir düzenlilikten çok, rastgele bir dağılım söz konusudur. Ancak bazı günlerde belirgin şekilde daha büyük depremler kaydedilmiştir. Örneğin, 3 Kasım ve 8 Kasım tarihlerinde büyük artışlar vardır.

Yoğunluk: Her tarih için çok sayıda küçük büyüklükte depremler (genellikle 1-2 ML arasında) kaydedilmiştir. Bu, bölgede sismik aktivitenin sürekli olduğunu ve küçük depremlerin sık yaşandığını göstermektedir.

Deprem Dinamiği: Türkiye gibi aktif fay hatlarının bulunduğu bölgelerde, küçük ve orta büyüklükteki depremler sık görülmektedir. Bu sismik aktivite, yer kabuğundaki gerilim birikimini göstermektedir ve bazen daha büyük depremler için bir işaret olabilmektedir.

Önlemler: Böyle bir aktivite, yapılaşma standartlarına uyulmasının önemini vurgulamaktadır. Özellikle sık ve ani artışlar gösteren bölgelerde, yapıların depreme dayanıklı inşa edilmesi kritik öneme sahiptir.

Deprem büyüklüğünün 5 veya daha fazlasına ulaşması potansiyel olarak yıkıcı olabilmektedir, ancak küçük artçı depremler genellikle ciddi zarar vermez. Bu grafikte gözlemlenen değişimler, sismik gözlem ve önlem almanın önemini hatırlatmaktadır.

Bu grafikte, deprem verilerinin üç boyutlu bir uzayda, enlem (x-ekseni), boylam (y-ekseni) ve derinlik (z-ekseni) boyutlarında görselleştirildiği bir dağılım göstermektedir. Her bir nokta, bir depremi temsil ederken, bu depremlerin konumu (enlem ve boylam) ve yerin ne kadar altında meydana geldiği (derinlik) gösterilmiştir.

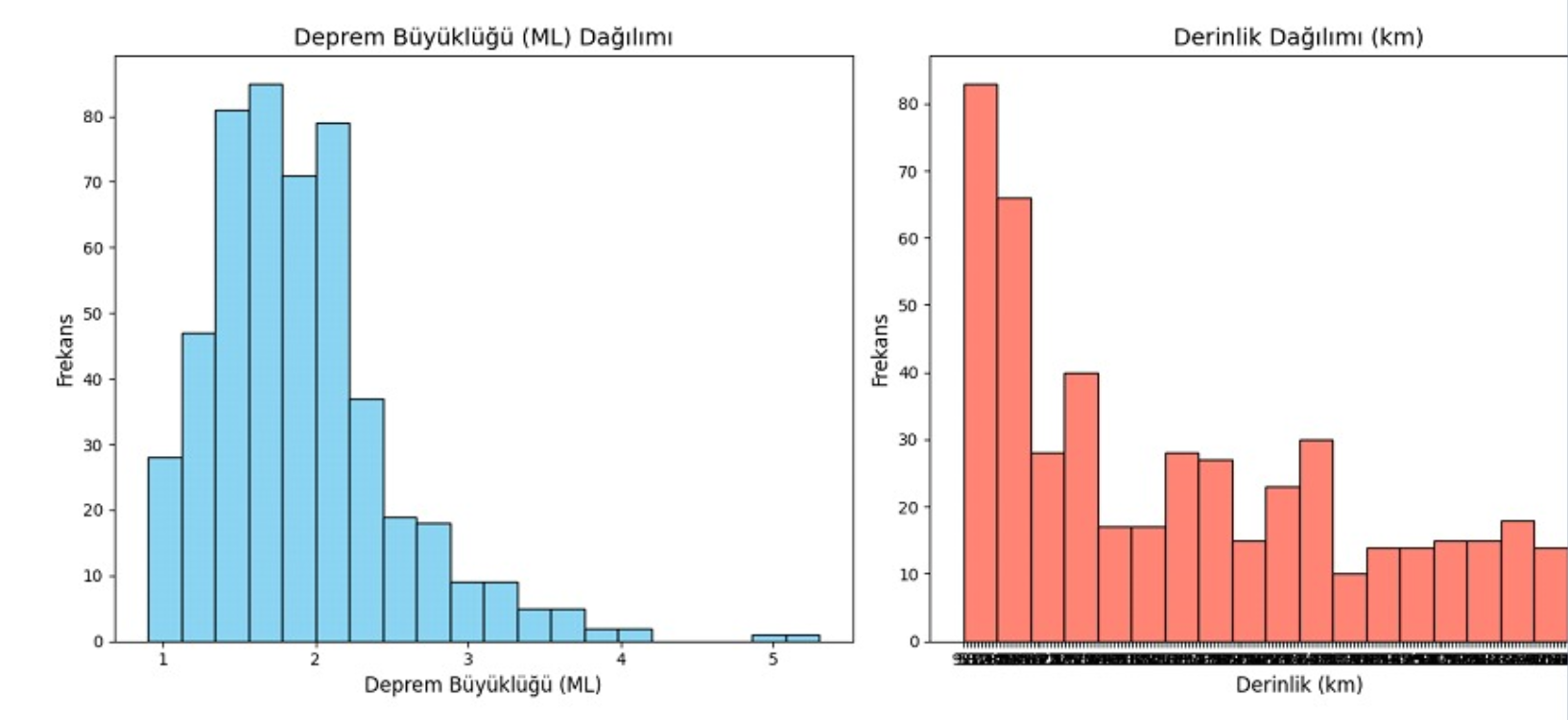
Yoğunluk ve Dağılım: Grafikteki yoğun noktalar, depremlerin belirli bir coğrafi bölge ve derinlikte daha sık meydana geldiğini göstermektedir. Depremlerin büyük çoğunluğu 0-20 km aralığında, yani sığ derinliklerde meydana gelmektedir. Bu, depremlerin yüzeye yakın yerlerde oluştuğunu ve bu nedenle potansiyel olarak daha fazla hasar riski taşıdığını işaret etmektedir.

Farklı Derinliklerdeki Depremler: Daha derin depremler (örneğin, 40 km ve daha derin) daha nadir ve izole olarak gözlemlenmiştir. Derin depremler genellikle daha geniş alanları etkiler ama yüzeydeki hasar potansiyelleri sığ depremlere kıyasla genellikle daha düşüktür.

Coğrafi Konumlar: Enlem ve boylam verileri, depremlerin belirli bir coğrafi alanda yoğunlaştığını göstermektedir. Türkiye’deki aktif fay hatları bu dağılımı açıklamaya yardımcı olmaktadır. Özellikle Kuzey Anadolu Fay Hattı ve Doğu Anadolu Fay Hattı gibi büyük fay sistemleri, depremlerin yoğunlaştığı bölgeler olarak bilinmektedir.

Sığ Depri emlerin Önemi: Sığ derinliklerdeki depremler, yüzeyde daha fazla hasar potansiyeline sahip oldukları için yapı güvenliği açısından büyük önem taşımaktadır. Bu nedenle, depreme dayanıklı yapı tasarımı ve zemin etütleri hayati öneme sahiptir.

Jeolojik Yapı: Türkiye, Alp-H malaya deprem kuşağında yer aldığından sık ve güçlü depremlerle karşılaşmaktadır. Bu grafikteki dağılım, bu jeolojik gerçeği destekler niteliktedir.

Bu grafikler, depremlerin büyüklük ve derinlik dağılımını görselleştiren histogramlardan oluşmaktadır. İlk grafik, depremlerin büyüklüklerine göre sıklıklarını, ikinci grafik ise derinliklerine göre sıklıklarını göstermektedir.

Deprem Büyüklüğü Dağılımı (Sol Grafik):

Büyüklük Aralığı: Depremlerin büyük çoğunluğu 1 ila 3 ML arasında yoğunlaşmış durumdadır. 2 ML civarında belirgin bir tepe noktasına ulaşan bu dağılım, çoğu depremin küçük büyüklüklerde meydana geldiğini göstermektedir. 4 ve 5 ML büyüklüğünde deprem sayısı ise oldukça düşüktür.

Depremlerin büyük bir kısmının düşük şiddette olması, bu depremlerin genellikle hasar yaratmadığını göstermektedir. Ancak, bu küçük depremler fay hatlarındaki enerji birikimini gösterdiği için dikkatle izlenmelidir. Bazen bu tür küçük depremler, daha büyük bir deprem için birikmiş enerjinin işareti olabilmektedir.

Derinlik Dağılımı (Sağ Grafik):

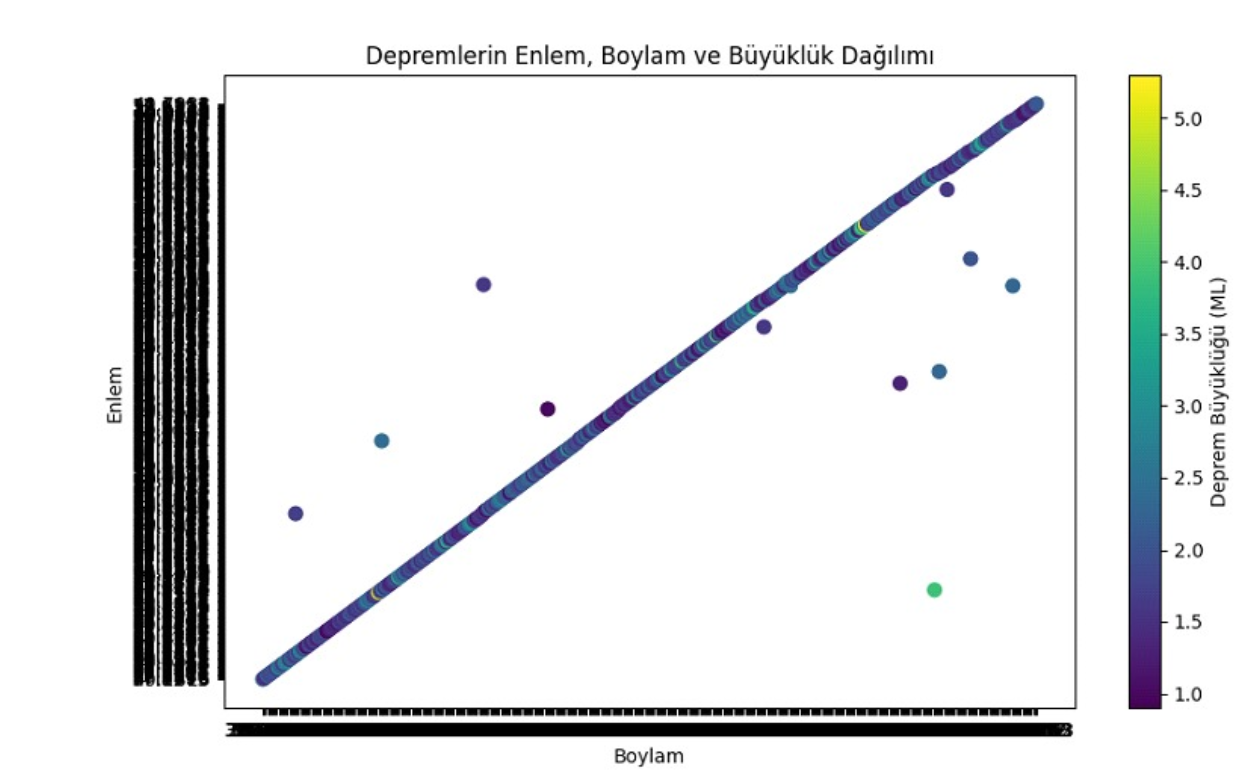
Sığ Depremler: Grafikte, 0-20 km aralığında meydana gelen depremlerin sayısı oldukça yüksek, bu da depremlerin genellikle yüzeye yakın olduğunu göstermektedir. 20 km’den daha derin olan depremler daha nadir ve daha geniş bir dağılıma sahiptir.

Sığ depremler, yüzeyde daha fazla hasar yaratma potansiyeline sahip oldukları için en çok endişe yaratan depremler arasındadır. Bu nedenle, bu bölgedeki yapılaşma, bu duruma uygun olarak planlanmaktadır.Derin depremler daha nadir olsa da, büyük ölçekli depremler genellikle daha derinlerde meydana gelmektedir.

Depremlerin Tehlike Potansiyeli: Veriler, çoğunlukla sığ ve küçük büyüklükteki depremlerden oluşsa da, her zaman daha büyük depremler için bir risk vardır. Bu nedenle bölgedeki sismik aktiviteyi sürekli izlemek önemlidir.

Deprem Bilgisi: Sığ derinliklerde meydana gelen küçük depremler genellikle normal fay hareketleriyle ilişkilidir ve stresin yavaşça salınmasına işaret eder. Ancak, uzun süreli stres birikimi büyük bir depreme yol açabilir.

Bu analiz, depremlerin büyüklük ve derinlik açısından nasıl dağıldığını ve bu dağılımın bölgenin sismik karakterini nasıl yansıttığını anlamamıza yardımcı olur. Fay hatları üzerinde yaşamanın getirdiği riskleri göz önüne alarak, yapı güvenliğine ve depreme hazırlık önlemlerine her zaman önem verilmelidir.

Deprem Yoğunluğu: Grafik, Türkiye'nin belirli bölgelerinde deprem aktivitesinin daha yoğun olduğunu göstermektedir. Bu bölgeler, genellikle aktif fay hatlarının bulunduğu alanlarla örtüşmektedir.

Büyüklük Dağılımı: Çoğu depremin orta büyüklükte olduğu görülmektedir. Ancak, zaman zaman daha büyük depremlerin de meydana geldiği anlaşılmaktadır.

Enlem-Boylam İlişkisi: Depremlerin enlem ve boylam değerlerine göre dağılımı, Türkiye'deki aktif fay hatlarının genel yönünü yansıtmaktadır.

Sismik Aktivite: Grafik, Türkiye'nin sismik olarak aktif bir bölge olduğunu bir kez daha teyit etmektedir.

Fay Hatları ile İlişki: Bu grafiği, Türkiye'deki bilinen fay hatları haritasıyla karşılaştırarak, depremlerin hangi fay hatları üzerinde yoğunlaştığını daha net görebiliriz. Bu, gelecekteki deprem tahminleri için önemli bir veri sağlayabilmektedir.

Deprem Sürüleri: Grafikte, belirli bölgelerde art arda meydana gelen deprem kümelenmeleri (sürüler) gözlemlenebilmektedir. Bu durum, o bölgedeki gerilme birikiminin artmakta olduğunu ve büyük bir depremin habercisi olabileceğini düşündürebilmektedir.

Deprem Derinliği: Grafikte depremlerin derinliği hakkında bilgi yer almamaktadır. Ancak, depremlerin derinliği, oluşan hasar ve hissedilme alanı açısından önemlidir. Bu nedenle, derinlik bilgisinin de eklenmesiyle analiz daha kapsamlı hale getirilebilir.

Zaman Faktörü: Bu grafik, belirli bir zaman aralığındaki depremleri göstermektedir. Farklı zaman aralıkları için oluşturulacak grafikler, deprem aktivitesindeki değişimleri ve uzun dönemli trendleri ortaya koyabilir.

SONUÇ

Bu raporda, Türkiye'nin sismik aktiviteleri analiz edilerek, ülkenin deprem riski altındaki durumunu ve bu durumun toplum üzerindeki etkilerini incelemeye çalıştık. Türkiye’nin konumu itibariyle birçok aktif fay hattına sahip olması, sürekli deprem riskini beraberinde getirmektedir. Yapılan analizler sonucunda, özellikle küçük ve orta büyüklükteki depremlerin sürekli olarak meydana geldiği, ancak zaman zaman büyük depremlerin de bu sürece eklendiği görülmüştür. Büyük depremler, fay hatlarında biriken enerji sonucunda gerçekleşen ani kırılmalarla ortaya çıkmakta ve önemli derecede hasara yol açabilmektedir.

Bu bağlamda, depremler üzerinde yapılan veri analizleri ve görselleştirmeler, sadece geçmiş olayları gözlemlemekle kalmayıp, gelecekteki olası depremler hakkında da bir öngörü sağlamaktadır. Özellikle yapay zeka ve veri bilimi uygulamaları ile geliştirilen modellerin, deprem tahminleri ve risk analizi gibi alanlarda kullanımı, daha güvenli bir gelecek için önemli bir adım teşkil etmektedir.

Depremlerin önlenemeyeceği gerçeğini kabul ederek, bu doğal afetlere karşı yapılabilecek en iyi hazırlığın, yapı güvenliği, toplum eğitimi ve erken uyarı sistemlerinin geliştirilmesi olduğu anlaşılmaktadır. Bu tür bilimsel analizler, toplum ve ilgili otoritelerin deprem farkındalığını artırarak, felaket riskini azaltmak adına daha bilinçli adımlar atılmasına katkı sağlamaktadır.