

# Veri Bilimi

## Otomatik Veri Çekme (Python)

---

M. Fuat Kına

<https://socialcomquant.ku.edu.tr/>



# Veri Bilimi



Veri oluşturma: çekme, toplama, sıfırdan yaratma



Temizleme ve önişleme




Veri analizi: model kurma, regresyon, sınıflandırma



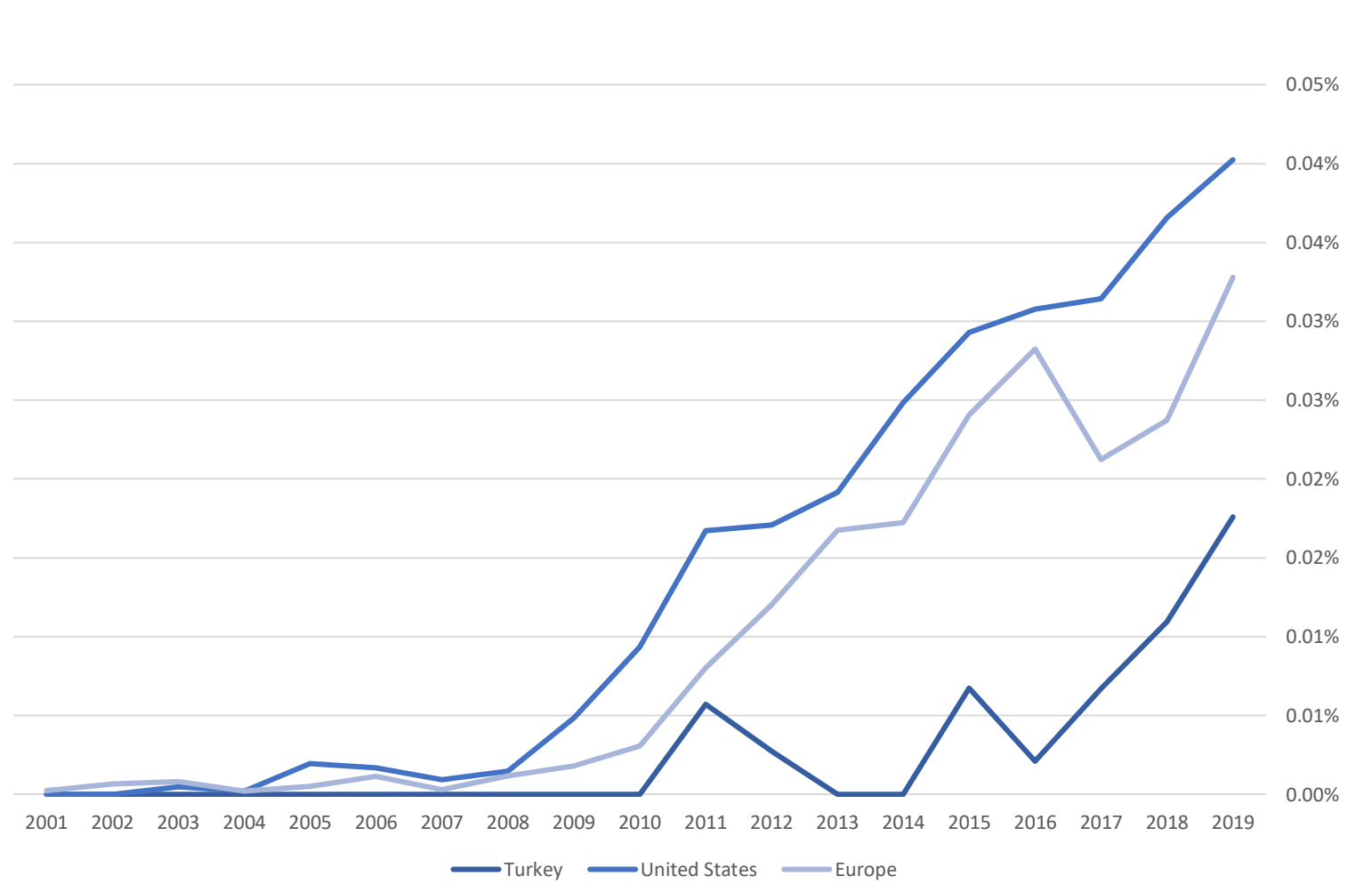
Modelin test edilmesi ve kullanılması: tahmin, optimizasyon, nedensellik, vb.

# Veri Bilimi Uygulama Alanları



- Sınıflandırma ve kategorizasyon
- Tahmine dayalı modelleme
- Duygu ve davranış analizi
- Öneri motorları ve kişiselleştirme sistemleri
- Konuşma sistemleri
- Anomali tespiti
- Kalıp (örüntü) tanımlama
- Otonom sistemler

## Veri Bilimi ve Sosyal Bilimler





# Hesaplamalı Sosyal Bilimler

- Metin Analizi (NLP)
- Coğrafi Bilgi Sistemleri
- Network Analizi
- Veri oluşturma (e.g. toplumsal hareketler, yoksulluk, göçmen karşıtlığı)

# Veri oluşturma



Hazır veri setleriyle çalışmak: World Bank, OECD, GLOCON, TÜİK, etc.



Uygulama programlama arayüzü (application programming interface): Sosyal medya, mesajlaşma uygulamaları



Veri kazıma



İleri veri oluşturma teknikleri: Yapay zeka, makine öğrenmesi, vb.

# Uygulama programlama arayüzü (API)

- API'lar Web sitesi operatörleri tarafından sağlanır.
- Erişim limitleri (kullanıcı onaylama, kullanım sınırları)

## Sosyal medya

- Facebook (<https://developers.facebook.com/>)
- Twitter (<https://developer.twitter.com/en/docs>)
- YouTube (<https://developers.google.com/youtube/v3>)
- Flickr (<https://www.flickr.com/services/api/>)
- Reddit (<https://www.reddit.com/dev/api/>)
- LinkedIn (<https://www.linkedin.com/developers/>)
- **Mesaj uygulamaları:** Telegram, WhatsApp, Threema, Skype, Discord
- **Akış uygulamaları:** Spotify, Apple Music, Vimeo, Twitch
- **Diğer servisler:** Google Maps, Amazon, Wikipedia

# Twitter API

- Kolay erişilebilir
- Zengin veri içeriği
- Retweet, favori, takipçi sayıları, tweet içeriği, biyografi, profil fotoğrafı, vb.
- Academic API, ayda 10 milyon tweet sağlıyor.

## Zorluklar:

- Gerçek insanların hesaplarından emin olmak zor
- Bu kişilerin kim olduğu hakkında bilgi eksikliği
- Platforma özgü limitler

Python'da ilgili kütüphane ([tweepy](#))



# Twitter API

```
# tweepy kütüphanesini çağırın
import tweepy as tw

# Twitter API key ve API secret bilgilerini tanımlayın
my_api_key = "MY_API_KEY"
my_api_secret = "MY_API_SECRET"

# kimlik doğrulama
auth = tw.OAuthHandler(my_api_key, my_api_secret)
api = tw.API(auth, wait_on_rate_limit=True)

search_query = "computational social science -filter:retweets"

tweets = api.search_tweets(q = search_query, lang = "en", count = 1000)
```

# Veri kazıma

- Sayfayı ve HTML kodunu didikle
- URL'yi (linki) tanımla, ve HTML sayfasını indir
- HTML kodlarını böl, parçala
- İlgili içeriği çıkart
- Veriyi yaz/kaydet

Python'da ilgili kütüphane ([scrapy](#))

# HTML yapısı

- html = elements + attributes + text
  - Hypertext markup language
  - Çocuk, ebeveyn, kardeş, gelecek kuşaklar...

```
<html>
  <body>
    <div id="div1" class="class-1">
      <p class="class-1 class-2">Hello World!</p>
      <p>Data Science!</p>
    </div>
  </body>
</html>
```

# Some HTML-Elements

## Metadata:

`<head>` collection of metadata  
`<title>` title of the document

## Sections:

`<body>` main contents of the document  
`<section>` section of the document  
`<h1>`, `<h2>`, ... headlines

## Grouped Contents

`<div>` container  
`<p>` paragraph

## Links

`<a>` Hyperlink, refers via href-attribute to resource

## Lists

`<ul>` unordered list  
`<ol>` ordered list  
`<li>` entry of a list

## Tables

`<table>` table  
`<tr>` row of a table  
`<td>` cell of a table

# Xpath

Html dosyasını Scrapy ile ayrıştırma

Sunumun bu noktadan sonraki kısmında yer alan kodlar için takip eden kaynaktan faydalandım:  
<https://app.datacamp.com/learn/courses/web-scraping-with-python>

- / → Bir nesil ileri git
- [] → Aradığımız elemanın sıralmasını belirt
- // → Tüm gelecek kuşaklar içinde ilerle

- *Xpath = '/html/body/div[2]//table'*

Head elemanını altındaki body elemanını altındaki ikinci div elemanının altındaki tüm tabloları seçer.

# Xpath

## Bazı örnekler:

- `'//p'`
- `'/head/body//div'`
- `'/div[3]//p'`

## Özellik seçimi:

- `Xpath = '//span[@class="span-class"]'`

Class özelliği `'span_class'` olan tüm span elemanlarını seçer.

# Xpath

**Örnek!** ‘Data Science’ kelime grubunu seç.

```
html =
'''<html>
  <body>
    <div>
      <p>Hello World!</p>
    <div>
      <p>Data Science!</p>
    </div>
  </div>
<div>
  <p>Thanks for Watching!</p>
</div>
</body>
</html>'''
```

Herhangi bir elemanın tüm çocuklarını seç:

- *Xpath* =  `'/html/body/*'`

Body elemanını altındaki tüm gelecek kuşakları seç:

- *Xpath* =  `"/html/body//*"`

HTML dökümanındaki tüm elemanları seç:

- *Xpath* =  `"//*"`

Artık \*, /, //, [], @ kullanarak, eleman veya nitelik arayarak ilgili yolu belirtebiliriz.

Not: Doğrudan eşleştirme yerine birden çok nesne içeren özellikler için (örneğin `<p class = 'class1 class2'>`) içerme fonksiyonunu (“contain”) kullanabiliriz.

# Xpath

## Örnek:

```
html = ""
<html>
  <body>
    <div id="div1" class="class-1">
      <p class="class-1 class-2">Hello World!</p>
      <div id="div2">
        <p id="p2" class="class-2">Choose
          <a href ="http://datascience.com">Data Science!</a>
        </p>
      </div>
    </div>
    <div id="div3" class="class-2">
      <p class="class-2">Thanks for Watching!</p>
    </div>
  </body>
</html>""
```

Id özelliğini kullanarak "Thanks for Watching!" ifadesini içeren paragraf ögesini seçelim

- *Xpath* = /html/body/div[2]/p

Class özelliğini kullanarak "Hello World!" ifadesini içeren paragraf ögesini seçelim

- *Xpath* = /html/body/div/p

"Data Science" hiper linki için ilgili bölümün href özelliğini seçelim (elemanın değil özelliğin içeriğini almak istediğimizi unutmayın)

- *Xpath* = /html/body/div/div/p/a/@href  
veya  
• *Xpath* = //a/@href



# Xpath ile metin çekme

```
<p id="p-example">  
  Hello world!  
  Try <a href="http://www.datacamp.com">DataCamp</a> today!  
</p>
```

- In XPath use `text()`

```
sel.xpath('//p[@id="p-example"]/text()').extract()  
# result: ['\n Hello world!\n Try ', ' today!\n']
```

```
sel.xpath('//p[@id="p-example"]//text()').extract()  
# result: ['\n Hello world!\n Try ', 'DataCamp', ' today!\n']
```

# Scrapy

```
# Scrapy kütüphanesinden Selector metodunu çağırın
from scrapy import Selector

# requests kütüphanesini indirin
import requests

url = 'https://m.imdb.com/chart/top'

# HTML kaynağını içeren html string'ini oluşturun
html = requests.get(url).content

# html objesini kullanarak bir Selector objesi tanımlayın
sel = Selector(text = html)

# ilgili xpath'i tanımlayın
xpath_for_movienames = '//h4/text()'

# ve aradığınız metin grubunu çekin!
sel.xpath(xpath_for_movienames).extract()
```

# Spider

```
import scrapy
from scrapy.crawler import CrawlerProcess

class IMDB_Spider(scrapy.Spider):
    name = "IMDB_Spider"

    def start_requests(self):
        yield scrapy.Request(url = 'https://m.imdb.com/chart/top',
                             callback = self.parse_front)

    def parse_front(self, response):

        movie_names = response.xpath('//h4/text()').extract()
```



Pratik uygulama: veri  
kazıma ve API kullanımı