UNIVERSITAS SULTAN AGENG TIRTAYASA

UNTIRTA Jawara
Jujur Adil Wibawa Amanah Religius Akuntabel

Data Mining

#9 Meeting

Web Scraping

Ferdian Bangkit Wijaya, S.Stat., M.Si
NIP. 19900520202406100

Investigation

1. Tahap Investigasi (Inspeksi Website)

Sebelum menulis kode, langkah paling krusial dalam web scraping adalah investigasi atau inspeksi halaman web target. Kita tidak bisa mengekstrak data yang tidak kita pahami strukturnya.

Tujuan:

- Memahami bagaimana data ditampilkan di halaman.
- Mengidentifikasi struktur HTML (tag, class, id) yang membungkus data yang kita inginkan.
- Mendeteksi apakah website bersifat statis atau dinamis.

Investigation

2. Cara Melakukan Investigasi:
   1. Gunakan "Inspect Element" (Developer Tools):
      - Klik kanan pada elemen data yang inginkan (misal, harga produk, judul berita) dan pilih "Inspect".
      - Perhatikan di panel Elements: Tag HTML apa yang digunakan (misal, , , <h1>)?
      - Apa nama class atau id unik yang bisa kita gunakan sebagai penanda?
   2. Bedakan Website Statis vs. Dinamis:
      - Statis: Jika "View Page Source" (Ctrl+U) dan data yang lihat di layar ada di dalam source code HTML tersebut, website itu statis.
      - Dinamis: Jika data tidak ada di Page Source (atau hanya terlihat seperti template JavaScript), data tersebut dimuat secara dinamis (menggunakan JavaScript) setelah halaman dibuka.
   3. Cek Network Tab (untuk API):
      - Buka Developer Tools, klik tab "Network", dan centang "Fetch/XHR".
      - Refresh halaman atau lakukan aksi (seperti scroll).
      - Jika melihat request ke suatu URL (misal, api.website.com/data) yang mengembalikan data dalam format JSON, ini adalah cara terbaik! Kita bisa langsung "menembak" API-nya.

Ekstraksi Statis

2.1. Ekstraksi Statis (Via Tag HTML/CSS Selector)

Metode ini digunakan untuk website statis. Data sudah ada di file HTML yang diunduh.

- Tools: requests (untuk mengunduh halaman) dan BeautifulSoup (untuk mem-parsing HTML).
- Logika:
    a. requests.get(url) untuk mengambil HTML.
    b. BeautifulSoup(response.text, 'html.parser') untuk mengubah HTML menjadi objek yang bisa dicari.
    c. Gunakan .find() atau .find_all() dengan penanda (tag, class, id) yang kita temukan di Tahap 1.

Ekstraksi Statis

Running di Python

Ekstraksi Dinamis

2.2. Ekstraksi Dinamis (Via Selenium)

Metode ini digunakan untuk website dinamis, di mana data dimuat oleh JavaScript (misalnya: infinite scroll, data yang muncul setelah klik tombol, dll).

- Tools: selenium (untuk mengotomatisasi browser) dan webdriver-manager (opsional, untuk manajemen driver).
- Logika:
  a. Selenium membuka browser sungguhan (misal, Chrome) secara otomatis.
  b. Browser tersebut memuat halaman, termasuk menjalankan JavaScript-nya.
  c. Kita bisa memberi perintah (seperti time.sleep()) untuk menunggu data muncul.
  d. Setelah data muncul, kita gunakan finder milik Selenium (misal, find_elements(By.CLASS_NAME, ...)).

Ekstraksi Dinamis

Running di Python

Ekstraksi via API

2.3. Ekstraksi Via API (Cara Paling Efisien)

Ini adalah metode terbaik jika tersedia. Kita tidak perlu mem-parsing HTML yang berantakan; kita langsung meminta data mentahnya ke server.

- Tools: requests (untuk mengambil data JSON).
- Logika:
  a. Temukan URL API di Tahap 1 (via tab Network).
  b. Gunakan requests.get() ke URL API tersebut.
  c. Data yang kembali biasanya dalam format JSON (JavaScript Object Notation).
  d. Gunakan response.json() untuk mengubah JSON menjadi Dictionary Python.
  e. Akses data seperti mengakses dictionary biasa.

Ekstraksi via API

1. API Publik (Contoh: PokeAPI, JSONPlaceholder)

Anggap ini seperti taman umum. Siapapun boleh masuk dan mengambil data (mengambil foto) kapan saja tanpa perlu tiket.

- Tujuan: Dibuat untuk developer berlatih, menguji kode, atau untuk materi demo seperti yang sedang kita siapkan.
- Ciri-ciri:
  - Tidak perlu registrasi.
  - Tidak perlu login atau "Token".
  - Bisa diakses langsung dari browser.
- Contoh: pokeapi, jsonplaceholder.typicode.com, API data COVID publik, beberapa API cuaca level dasar.

Ekstraksi via API

2. API Terproteksi

Ini adalah mayoritas API di dunia nyata. Anggap ini seperti bioskop atau konser. Harus punya tiket (Token/API Key) untuk bisa masuk.

- Tujuan:
  - Keamanan: Memastikan hanya pengguna terdaftar yang bisa ambil data.
  - Monetisasi: Perusahaan bisa menjual akses ke datanya (misal: API Google Maps).
  - Kontrol (Rate Limiting): Mencegah 1 orang melakukan scraping jutaan data per detik yang bisa membuat server down.
- Ciri-ciri:
  - Harus mendaftar di dashboard developer website tersebut.
  - Akan diberi API Key atau Token (seperti Bearer Token).
  - Token ini harus disertakan dalam setiap request yang kita kirim.

Ekstraksi via API

Running di Python

Rapikan Data

Data yang didapat dari scraping (terutama metode 1 dan 2) hampir selalu kotor. Data tersebut perlu dibersihkan sebelum dianalisis atau dimasukkan ke database.

Tugas Umum Pembersihan:

- Menghapus spasi putih berlebih (strip()).
- Menghapus karakter aneh (misal, \n, \t, atau "Rp.", "kg").
- Mengubah tipe data (misal, dari string "1.000" menjadi integer 1000).
- Menstandardisasi format (misal, "Jkt" -> "Jakarta").
- Tools: pandas (paling umum digunakan).

Rapikan Data

Running di Python

UNIVERSITAS SULTAN AGENG TIRTAYASA

UNTIRTA Jawara
Jujur Adil Wibawa Amanah Religius Akuntabel

SEE YOU NEXT WEEK !

Ferdian Bangkit Wijaya, S.Stat., M.Si
NIP. 19900520202406100 1
ferdian.bangkit@untirta.ac.id